

U-155
Letter

March 17, 1987

NO ITEM TO INSERT

NO ITEM TO INSERT

Dear

NO ITEM TO INSERT

:

Systems development productivity tools, ranging from applications development tools, software maintenance tools, and fourth/fifth generation languages to data base management systems, have been and continue to be developed. The quality, variety, and use of such tools have all increased over the past five years, but there is serious doubt as to whether hardware/software performance has improved.

This is primarily due to two things: productivity is only being targeted at the code/language/data base level rather than at the systems level and little or no attention is being paid to the quality and use of information. The emphasis is on code and data production rates rather than whether the data is useful or the code is efficient.

INPUT's report, Software Productivity, addresses these issues and is enclosed for your review. Please call us if you have any queries or concerns.

Yours sincerely,

Graham S. Kemp
Vice President

GSK:ml

Enclosure



SOFTWARE PRODUCTIVITY

DECEMBER 1986



Published by
INPUT
1943 Landings Drive
Mountain View, CA 94043
U.S.A.

Information Systems Program (ISP)

Software Productivity

Copyright ©1986 by INPUT. All rights reserved.
Printed in the United States of America.

No part of this publication may be reproduced or
distributed in any form or by any means, or stored
in a data base or retrieval system, without the prior
written permission of the publisher.

INPUT

SOFTWARE PRODUCTIVITY

ABSTRACT

Systems development productivity tools, ranging from applications development tools, software maintenance tools, and fourth/fifth generation languages to data base management systems, have been and continue to be developed. The quality, variety, and use of such tools have all increased over the past five years, but there is serious doubt as to whether hardware/software performance has improved.

This is primarily due to two things: productivity is only being targeted at the code/language/data base level rather than at the systems level, and little or no attention is being paid to the quality and use of information, e.g., the emphasis is on code and data production rates rather than whether the data is useful or the code is efficient.

This report contains 148 pages, including 26 exhibits.





SOFTWARE PRODUCTIVITY

CONTENTS

	<u>Page</u>
I INTRODUCTION.....	1
A. Objectives, Audience, and Need	1
B. Scope and Use	3
C. Methodology	4
D. Related INPUT Reports	6
II EXECUTIVE SUMMARY	7
A. Solutions--A Communications Gap	8
B. Upside Down	10
C. Backwards	12
D. Productivity/Performance/Problems	14
E. Productivity Plan	16
F. Requirements By Performance Level	18
G. Recommended Changes of Direction	20
III THE PROBLEM DEFINED	23
A. Summary of Past INPUT Findings	23
B. Problems versus Solutions	32
1. Systems Complexity	33
2. Matching Problems to Solutions	36
C. Solutions versus Problems	37
1. Matching Solutions to Problems	38
2. Solutions Becoming Problems	40
D. Factory versus Office	41
1. The Measurement Problem	42
2. Performance Measurement Defined	44
a. Blue Collar Workers	44
b. White Collar Workers	45
E. Data/Information/Knowledge Quality	46
1. Practical Definitions	47
2. Beyond Semantics	49
IV THE SOLUTIONS EVALUATED	51
A. Conventional Approaches	51
1. Languages	52
2. Other Tools, Aids, and Methodologies	63
B. Distributed Systems Development	73
C. Case Studies	93
1. Case Study #1	94
2. Case Study #2	99



	<u>Page</u>
3. Case Study #3	104
4. Case Study #4	109
5. Case Study #5	112
V FUTURE DIRECTIONS.....	117
A. Integrated Applications Development Systems	117
B. Network Evolution	122
C. Media Revolution	126
D. AI and All That Implies	127
E. The Data/Information/Knowledge Model	131
F. The Users' View of Future Productivity Improvement	135
VI CONCLUSIONS AND RECOMMENDATIONS	139
A. Conclusions	139
B. Recommendations	147



SOFTWARE PRODUCTIVITY

EXHIBITS

		<u>Page</u>
II	-1 Solutions--A Communications Gap	9
	-2 The Productivity Pyramid, 1980	11
	-3 Backwards--Computer/Communications Networks	13
	-4 Productivity/Performance/Problems	15
	-5 Productivity Plan	17
	-6 Requirements By Performance Level	19
	-7 Recommended Changes of Direction	21
III	-1 Time Distribution - 1964/1980/1986	26
	-2 The Productivity Pyramid, 1980	28
	-3 Ranges of Programming Performance	30
	-4 Distribution of Office Workers' Time	43
IV	-1 A Schematic for Evaluation	53
	-2 Primary Languages Used	56
	-3 Fourth Generation Languages Installed	57
	-4 Savings from Higher Level Languages	62
	-5 Ratings of Importance of Productivity Tools, Aids, and Methodologies	65
	-6 Estimated Savings on Productivity Tools	68
	-7 Opinions Concerning Performance Improvement	70
	-8 Ratings of DSD Problems	79
	-9 Effective Approaches to DSD	81
	-10 Backlog Analysis	84
	-11 Maintenance	87
	-12 Performance Improvement and DSD	90
V	-1 IBM's Preferred Solution	124
	-2 The Data/Information/Knowledge Model	132
	-3 Future Expectations	136





I INTRODUCTION





I INTRODUCTION

A. OBJECTIVES, AUDIENCE, AND NEED

- It is currently popular to say that systems developers have "automated" many business functions but have not applied computer technology in automating the systems development process itself. This statement is wrong. The systems development process is the most automated of any office function, and an inordinate percentage of the total systems development effort (and computer processing power) has gone into automating it. The important fact is that, despite our best efforts, productivity in computer systems development remains a major problem. One of the primary objectives of this study will be to define this problem.
- That means that we are going to take a systems approach in analyzing what is needed to improve productivity in the applications development process. Simply stated, the objective of this report is to define the requirements for improving productivity among those employees whose primary responsibility is the development of computer applications systems. While this can be simply stated, it is complicated by a fact which all systems developers know-- requirements have a way of changing before the system (solution) can be developed. Therefore, this report will concentrate on what is needed to improve productivity in both the current and anticipated hardware/software systems environment.



- In addition, in the firm belief that it is impossible to improve anything (including productivity) unless you can measure it, an additional objective of this study will be to define a comprehensive framework within which office productivity (including that of those employees involved in the systems development process) can be measured. By providing a framework for performance measurement, evaluation of current tools, aids, and approaches to productivity improvement will be possible and a foundation for cost justification will be established.
- This report is essentially a planning document and is directed specifically toward the vice president of information systems in user organizations and the vice president of planning for companies engaged in the development of productivity tools, aids, and approaches. In addition, it should be of interest to those actively engaged at the development level in both user and vendor organizations.
- The need for this report is manifest in the continued failure of past solutions to solve the productivity problem and the seeming reluctance on the part of many IS departments to employ the tools, aids, and approaches which are currently available. Users have long been aware that there is a tremendous gap between current user needs and existing productivity tools. This problem has been compounded by the current mass marketing of software productivity "solutions" at all levels in the processing hierarchy—yesterday's solutions (departmental systems and personal computers) rapidly become today's problem (connectivity). This report attempts to stand back and take a longer range view of the productivity problem based on the anticipated operating environment.



B. SCOPE AND USE

- This report will address all aspects of the productivity problem and attempt to put the contribution of tools and aids into proper perspective. It will use the following INPUT systems categories for this purpose:
 - Productivity hierarchy.
 - Commitment to quality.
 - End-user involvement.
 - Broadbased management.
 - High quality personnel.
 - Tools/aids/methodologies.
 - Performance.
 - Hardware/software.
 - Human/machine dyad.
 - Work unit.
 - Institutional.
- The report will cover the obvious trends toward applications development systems and integrated development environments. However, the emphasis will be on projected requirements for developing distributed systems. Essentially, this analysis concerns itself with the problems and opportunities



of an environment in which both development activities and the resulting applications systems will be less centralized and integration, differentiation, and mechanization will become increasingly important.

- Detailed functional analysis of existing tools and aids is beyond the scope of this study. However, the emphasis on requirements will afford an ample opportunity for generic evaluation of such products.
- For computer/communications users, it can be used as a fairly comprehensive guide for developing a plan for productivity improvement. In addition, the comprehensive framework provided for evaluation of productivity improvement can serve as a useful guide in preparing cost justifications for improved productivity plans and for the tools and aids necessary to implement such plans.
- For vendors, this study, with its emphasis on user requirements, will serve as an effective guide for product development and enhancement. In many ways, the report is designed to be a communications link between users and vendors.

C. METHODOLOGY

- INPUT's extensive research activities in productivity improvement has provided us with a data base against which deviations can be identified quite readily. Thirty telephone interviews were conducted to verify our existing data base which is based on approximately 2,000 productivity interviews conducted over the years.
- Five comprehensive case studies were conducted. Three of these involved on-site interviews with organizations which were also used as case studies this year for the INPUT reports on operating systems and departmental software. The other two placed emphasis on two important aspects of productivity



improvement--the continuing problem of maintenance and the use of advanced applications development systems.

- Fifteen senior interviews were conducted by the project leader. These interviews fall into the following categories:
 - Senior IS management who are on the leading edge in five specific industries (transportation, education, insurance, banking, and manufacturing) were interviewed concerning their productivity improvement plans and as a means of verifying INPUT's conclusions on productivity. These latest interviews represent a continuing dialog on productivity improvement with these executives which extends back over five years.
 - Five recognized leaders in fields significant to software productivity improvement were interviewed. These "experts" were interviewed on the following topics--DBMS (including associated languages), applications generators, artificial intelligence, IBM's activities in productivity improvement, and operations research (with special emphasis on queuing networks).
 - Senior management of five companies on the leading edge in the development of applications development systems (ADS) were interviewed concerning their products and future directions in productivity.
- In addition, less structured interviews were conducted with approximately 20 individuals who have contributed substantially to past INPUT research efforts in the area of productivity improvement in the software development process.
- It is INPUT's opinion that any research in complex areas such as productivity improvement must be supported by extensive secondary research, and because of our interest in this area, such research is conducted on a continuing basis. The result has been a substantial research base on this particular subject.



D. RELATED INPUT REPORTS

- The following INPUT reports have served as the foundation for this current study and will provide substantially more support for some of the conclusions which are reached in this report:
 - Systems and Software Productivity, 1980.
 - Software Development Productivity, 1982.
 - Impact of Office Systems on Productivity, 1983.
 - Relational Data Base Developments, 1983.
 - Market Impacts of IBM Software Strategies, 1984.
 - Market Impact of New Software Productivity Techniques, 1984.
 - New Opportunities for Software Productivity Improvement, 1984.
 - Artificial Intelligence and Expert Systems, 1985.
 - Market Analysis: Data Base Management Systems, 1985.
 - Market Analysis: Fourth Generation Languages, 1985.
 - Market Analysis: Applications Development Tools, 1985.



II EXECUTIVE SUMMARY







II EXECUTIVE SUMMARY

- This executive summary is designed in a presentation format in order to:
 - Help the busy reader quickly review key research findings.
 - Provide an executive presentation and script that facilitates group communications.
- The key points of the report are summarized in Exhibits II-1 through II-7. On the left-hand page facing each exhibit is a script explaining the exhibit's contents.



A. SOLUTIONS—A COMMUNICATIONS GAP

- The solutions to the productivity problem are quite different for the development staff and for end users.
 - The development staff has traditionally depended on computer languages and data base management systems to improve productivity in developing computer applications. Currently, the emphasis is on 4GLs and relational DBMSs.
 - The primary productivity tools of end users are word processing packages and spreadsheets. While DBMSs are included in integrated packages, users do not utilize them for any significant portion of their work. The impact of user productivity tools has primarily been on calculators and typewriters.
- The major problem continues to be a significant communications gap between the development staff and end users. The development staff feels the end users do not understand the complexity of what they are asking for and in any case must be controlled by standards, access hierarchies, and security, while the end users ask only for data so they can do what they want with it. There is a major conflict between top-down versus bottom-up systems design, and there is chaos in computer/communications networking, especially at the departmental level.
- It appears apparent that there is currently little reason to believe that the central development staff with its large mainframe orientation and the end users with their PCs are developing applications which can be effectively integrated into systems that will be of maximum benefit to their common company or organization.



SOLUTIONS - A COMMUNICATIONS GAP

	DEVELOPMENT STAFF	END USERS
Productivity Tools Required	<ul style="list-style-type: none">● 4GLs● Relational DBMS● Etc.	<ul style="list-style-type: none">● Word Processing● Spread-sheets● Etc.
Design Approach	Top-Down	Bottom-Up
Need	Control	Data

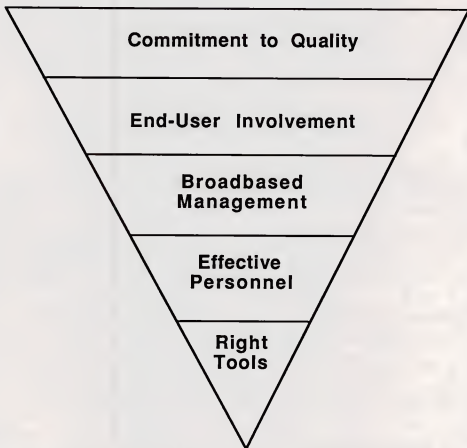


B. UPSIDE DOWN

- Past INPUT research into productivity has indicated that, in order to have a truly productive environment for developing systems, it is necessary to establish the following priority sequence:
 - One, there must be a commitment to quality.
 - Two, end users must be involved in the development process.
 - Three, there must be broadbased management of development projects.
 - Four, effective personnel must be assigned to the project.
 - Five, the right tools must be selected based on both the nature of the project and the personnel who have been assigned.
- Past research disclosed (and current research confirms) that primary emphasis is being placed on tools and little attention is being given to quality. The "productivity pyramid" has been turned upside down by the "distributed systems development" environment which has been created by the use of PCs, micro/mainframe links, information centers, prototyping, and the general confusion concerning networking and "connectivity."
- The typical "solutions" attempted therefore contribute to the problem in this topsy-turvy environment. They are essentially short-term solutions with long-term impacts, which do not focus on either quality or end-user involvement and which ultimately add to the list of long-term concerns to be resolved.



**THE PRODUCTIVITY PYRAMID
1980**

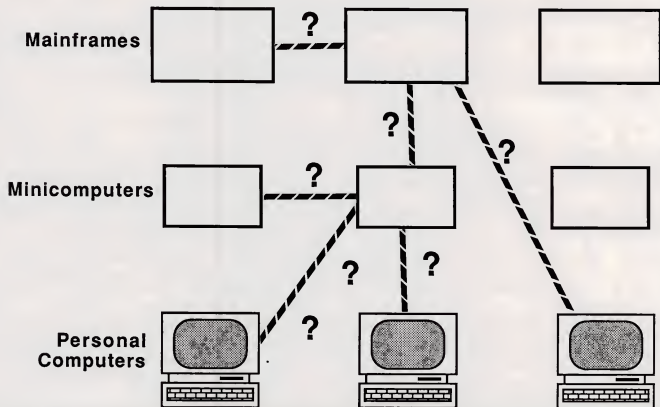




C. BACKWARDS

- Besides being "upside down" in our approach to productivity improvement, there are those who believe that we are going about computer/communications network development "backwards."
- A prominent computer industry executive has been quoted as stating that we have been literally going about networking backwards by "buying a lot of computers and then trying to tie them together." The solution recommended was to "build the network first and hang the computers on later."
- This type of reasoning fundamentally says that rather than concentrating on standalone and/or loosely coupled data processing applications, the emphasis should be on information flow between and among humans, organizations, and computers. It is difficult to argue with this bit of wisdom, and one of the case study companies in this study seems to have had substantial success by concentrating on network development and worrying about specific applications later.
- This type of approach is foreign to most central IS departments which are large mainframe, central data base-oriented in their approach to systems development. Going about network development in a straightforward manner has not been characteristic of either vendors or those responsible for computer systems development.



INPUT[®]**BACKWARDS
COMPUTER/COMMUNICATIONS NETWORKS**



D. PRODUCTIVITY/PERFORMANCE/PROBLEMS

- INPUT believes that true productivity (either of the enterprise or in the systems development process) must be measured by performance at four levels:
 - The hardware/software level which includes the cost of all hardware and software on both an investment and ongoing (operational) basis and the throughput of the system in terms of productive work.
 - The human/machine dyad which measures the combined cost of the human and machine and the resulting output the dyad is able to achieve.
 - The work unit, an organizational (rather than geographic) entity, which includes the cost of interpersonal communications and overhead activities.
 - The institutional level which can be the classic "bottom line" or other suitable measure of achieving goals and objectives in a cost-effective manner.
- These levels are interrelated, but maximization at one level does not necessarily have positive impact on the other. (For example, lines of code or quantity of paper produced at the human/machine dyad may or may not have positive impacts on the other performance levels.)



INPUT[®]

PRODUCTIVITY/PERFORMANCE/PROBLEMS

	PERFORMANCE LEVEL	IMPACT OF TOOLS
I	Hardware/Software	Negative
II	Human/Machine	Positive
III	Work Unit	Negative
IV	Institutional	Unclear



E. PRODUCTIVITY PLAN

- It is INPUT's conclusion that the IS function needs a productivity plan which rights the productivity pyramid by establishing priorities that emphasize quality and performance at all four performance levels. The answer to improved productivity is not throwing more hardware and software at business problems and assuming that computerized solutions are the total answer. Quick and dirty systems development in order to meet schedules and turn projects over for maintenance are counterproductive.
- It is necessary to get end users involved during all phases of the systems life cycle for all major projects and not view end-user computing as a convenient way of keeping down end-user demands while the development teams work on the really important projects. The active participation of both user and executive management in all phases of major development projects should be encouraged, and both end users and management should share the commitment to quality which is the foundation of any productivity improvement plan.
- The attraction, motivation, management, and retention of effective personnel should be of primary concern. Most competent IS management recognizes that throwing bodies at productivity problems is counterproductive and can actually take longer and produce inferior systems. The temptation to constantly grow the organization is not necessarily an integral part of a good productivity plan.
- The right tools to establish a truly productive environment become secondary if attention is given to the more fundamental aspects of a productivity improvement program. There is no shortage of good tools, but the quest for a magical solution to the entire productivity problem can result in substantial wasted effort.



PRODUCTIVITY PLAN

- **Establish Priorities Emphasizing Quality and Performance**
 - **Active Executive Management and End-User Involvement**
 - **Motivation, Management, and Retention of Effective Personnel**
 - **Right Tools Secondary**
-



F. REQUIREMENTS BY PERFORMANCE LEVEL

- The IS department must concern itself more with the use and quality of data and information at the various performance levels which contribute to a productive environment. At the hardware/software level, more attention must be given to performance monitoring and the impact of the tools used to develop systems on the operational characteristics of those systems. The IS function must accept responsibility for establishing a productive hardware/software environment and not become overly dependent on the current solutions provided by outside vendors.
- At the human/machine dyad level, the IS function has a responsibility to provide education and training in the effective use of the tools chosen. The first thing which will be necessary is to convince PC users that their PC tools are not "applications" and that there are elements of both programming and data base management disciplines which must be applied when using them.
- At the work unit level, IS must become familiar with the company flow of information (mostly paper systems and procedures) and help users understand the quality of the data and information they receive from the central computer facility. IS must provide leadership in educating work units on systems concepts and in the major technological change from paper to electronic media.
- At the institutional level, data, information, and knowledge must be understood and qualified in terms of content, integrity, and use. The ability to recognize the difference between information and knowledge is of primary importance. Before building knowledge-based systems, it is necessary to identify knowledgeable people.



REQUIREMENTS BY PERFORMANCE LEVEL

- **Hardware/Software**
 - Performance Monitoring
 - Privacy and Security
 - Environment Productive Hardware/Software
 - **Human/Machine Dyad**
 - Education and Training
 - Programming and Systems Concepts
 - **Work Unit**
 - Quality Control Systems
 - Systems Concepts
 - Media Replacement (Paper → Electronic)
 - **Institutional**
 - Data/Information/Knowledge Content, Integrity, and Flexibility
 - Knowledge Identification
-



G. RECOMMENDED CHANGES OF DIRECTION

- INPUT recommends that the IS function broaden the scope of its vision and activities.
 - Emphasis must be shifted from data processing (computer) applications to information flow within the organization.
 - Productivity must be measured not by the quantity of data/information produced but by the quality.
 - Rather than automate current office processes, the processes themselves must be improved and better understood.
 - Gradually, the emphasis on information must give way to the identification of knowledge and the information which is necessary to improve and create new knowledge.
 - The IS function must change from being application builders to becoming data/information/knowledge architects (which is another way of saying that systems personnel must understand the business they are in).
 - The whole purpose of computer systems is to improve productivity, and the systems developers must become productivity consultants to management in the broadest sense of the term; in other words, at all four performance levels.



RECOMMENDED CHANGES OF DIRECTION

- Data Processing Applications → Information Flow
 - Information Quantity → Information Quality
 - Automation of Process → Improved Process
 - Information Emphasis → Knowledge Emphasis
 - Application Builders → D/I/K Architects
 - Systems Developers → Productivity Consultants
-





III THE PROBLEM DEFINED





III THE PROBLEM DEFINED

A. SUMMARY OF PAST INPUT FINDINGS

- Nearly 25 years ago IBM conducted a comprehensive survey of all its large-scale computer users. At that time, there were approximately 500 large-scale, commercial IBM computer installations in the U.S. Over 80% of those installations responded to the survey and revealed the following:
 - In excess of 90% of all program libraries were written in Autocoder.
 - Approximately 85% of the respondents stated they would not convert these production programs to a higher level language.
 - Over 95% of the respondents were using Autocoder and stated they would continue to use it.
 - Less than 30% of all respondents were using COBOL, and slightly more than 50% stated categorically that they would not use it.
 - While most respondents did not have any knowledge of COBOL, of the ones who did, both speed and quality were rated "poor" twice as often as they were "good."



- These same respondents overwhelmingly voted for "ease of use" as the most important attribute of all languages (assemblers, algebraic, English, and RPGs).
- It was concluded that both programmers and organizations do not like to change languages. All INPUT research confirms the natural reluctance on the part of both organizations and individuals to change either languages or the way systems are developed.
- These Autocoder programmers were still handling card decks, suffering with turnaround problems (overnight was considered good in most installations), and having to deal directly with rather complex input/output control systems (IOCS) while defining data files for each specific application. In addition, many of the applications were being "computerized" for the first time. It was a time of trial and error, and the infamous "spaghetti code" was the rule of the day. Those developing systems in this manner reported they spent their time as follows:
 - 44% of their time was spent on requirements analysis and systems design.
 - 20.3% was spent on "coding" (programming).
 - 19.1% was spent on testing and debugging.
 - 16.6% was spent on documenting and installing the system.
- In 1980, INPUT conducted one the most comprehensive studies of productivity in the systems development process which has ever been made. One of the clients for that study was a university which has pioneered in providing productivity tools and in creating a highly productive environment for systems development. (That institution is currently a case study location for INPUT research.) Interactive computing for IBM mainframes was pioneered at this



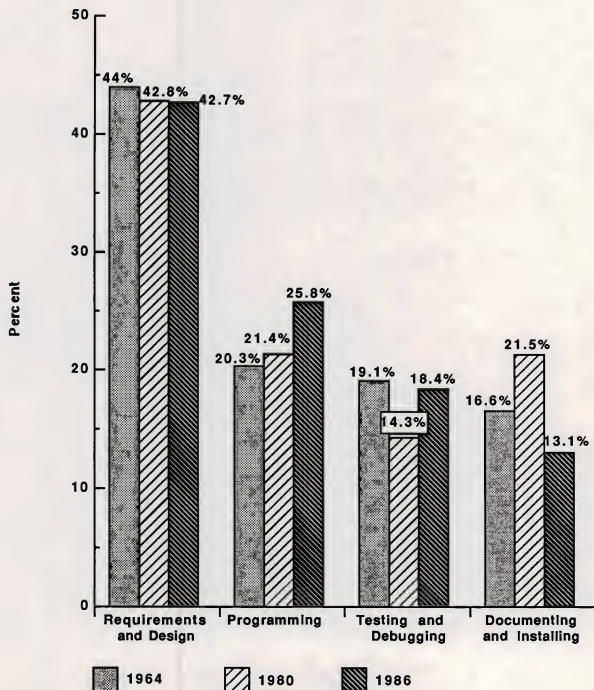
university, all systems developers had terminals at their desks, and many were encouraged to have them at home also. Major operating systems enhancements had been made to assure high response and provide features IBM is still struggling with. A proprietary DBMS and associated higher level language had been developed and established as a standard for applications development. An extensive electronic mail system had been established throughout the campus. The development group had excellent records on how development projects broke down in terms of effort. The distribution of time was as follows:

- 42.8% was spent on requirements analysis and systems design.
 - 21.4% was spent on programming.
 - 14.3% was spent on testing and debugging.
 - 21.5% was spent on documenting and installing systems.
- We found these numbers to be surprisingly similar to those which were reported by the general population nearly 20 years before. The similarity led to considerable analysis and some conclusions which will be reported later. However, since 1980 personal computers have come on the scene and established new standards for ease of use, 4GLs and relational DBMSs have been embraced, information centers have become accepted, and prototyping has become commonly accepted by many companies. Tools, aids, and approaches to improved productivity in the systems development process are being announced on practically a daily basis. What has been the impact on how time is spent in the systems development process? The time distribution reported by respondents in the research for this study are presented in Exhibit III-1.
- Requirements analysis and systems design continue to account for over 40% of the systems development effort at 42.7%.



EXHIBIT III-1

TIME DISTRIBUTION - 1964/1980/1986





- Programming, where most of the tools and aids have been directed in the past, has actually increased to 25.8%, up 27% from the 20.3% in 1963.
 - Testing and debugging, despite structured methodologies' assault on "spaghetti code," still requires 18.4% of the development effort.
 - Documentation and installation have fallen to 13% of the development effort, but the total effort from the time programming starts to installation remains slightly higher than it was in 1963 (58.3% as compared to 56%).
-
- While numerous explanations and rationalizations of this rather remarkable similarity in time distribution can be made, the fact remains that there seems to be something almost "God-given" about what is required when human beings try to communicate what they want computers to do for them. The "programming problem" has withstood "solutions" which have been the products of some of the most fertile and creative human minds we have to offer, and there is little reason to feel optimistic that the development process will be automated in the foreseeable future.
 - In fact, all past INPUT analysis of the problem has led to the unmistakable conclusion that the continuing efforts to automate the systems development process have tended to obscure other more important factors which are essential for improving productivity of systems personnel. This conclusion led to the identification of what is referred to as the productivity pyramid (see Exhibit III-2) which not only depicts the relative importance of the factors contributing to improved productivity but also provides a means of identifying the requirements for a truly productive environment.
 - The foundation of the productivity pyramid is an underlying commitment to the quality of the system being developed. If we have learned anything in the last 20 years, it is that "quick and dirty" systems development costs more over the systems life cycle.



EXHIBIT III-2

**THE PRODUCTIVITY PYRAMID
1980**





- It has also been learned that "commitment" is nice, but unless end users are involved in the development process early and on a continuing basis, it is impossible to develop a high-quality system.
- Today's computer/communications systems cut across organizational lines for development, use, data access, and operation. It is absolutely essential to get general management of all affected organizations involved in establishing priorities, monitoring schedules, and commitment to the project. In other words, broadbased management implies acknowledging and accepting responsibility for the success or failure of the project.
- The definition of effective personnel to implement the system will vary depending on the specific application. It must be recognized that the capabilities of systems personnel extend over a very wide range, a range which far exceeds the potential impact of any tools and aids which are employed in implementation (see Exhibit III-3). These ranges of observed individual performance should be understood because of some of the later conclusions which are reached concerning the productivity problem. What these numbers say is this:
 - By using a simple measure such as the number of lines of code produced, some individuals will produce five times as much "work."
 - Given the task of removing defects from an existing application, some programmers will take ten times as long as others.
 - The same application developed using the same tools by two individuals may require as much as 11 times the machine resources to develop.



EXHIBIT III-3

RANGES OF PROGRAMMING PERFORMANCE

Distribution
of Systems
Population

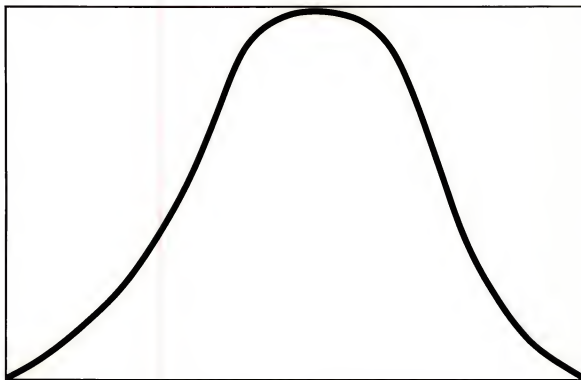
F

D

C

B

A



Defeat
Renewal
10-1

30-1
Debugging

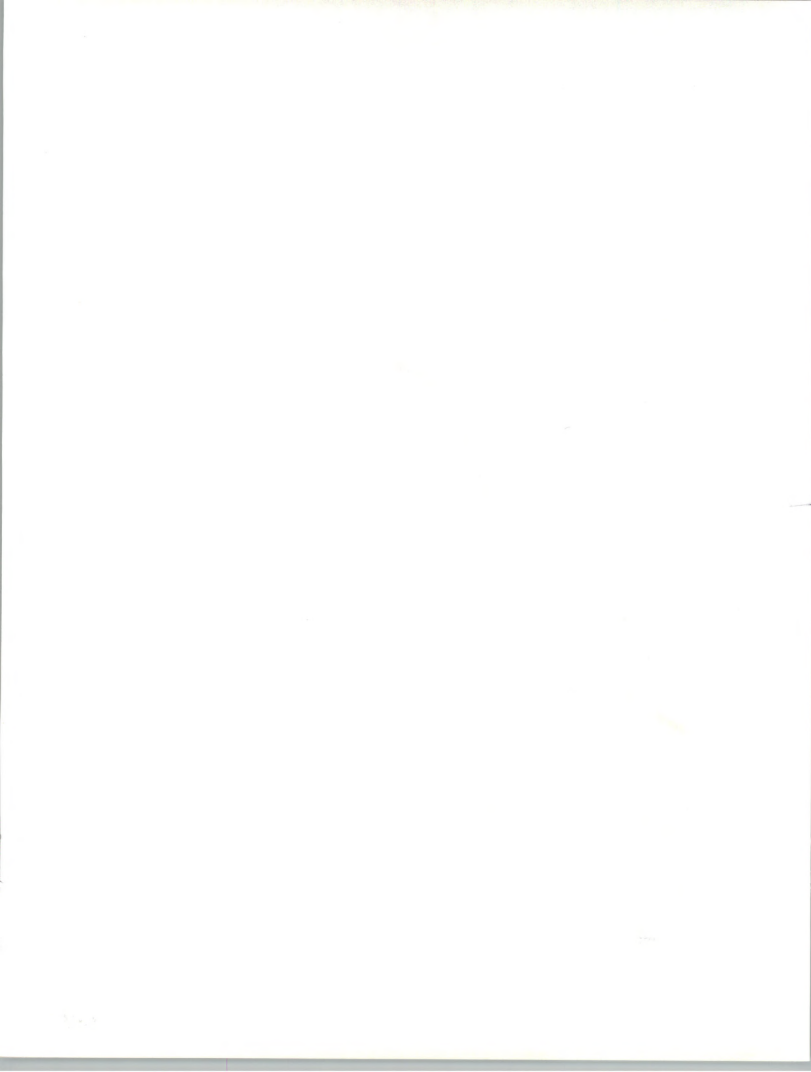
Line of Code
5-1

25-1
Coding

Range of
Capability

13-1
Execution Speed

11-1



- The continuing cost of running the application may be up to 13 times as much.
- The programmer's time to get the application running may require one programmer up to 25 times as long as another.
- Debugging of an application may require up to 30 times as long for some individuals.
- Actually, on the latter two categories some observers insist the range is infinite since some individuals would never get certain applications running. There is something to be said for this point of view when considering the assignment of effective personnel to projects.
- Based on considerable experience with aptitude testing for systems work, it has been reasonably well established that the available supply of candidates for systems work follows a normal distribution based on test scores. (The A through F scale is that which was established for the Programmers Aptitude Test (PAT) in the 1960s.)
- In addition, training and experience seem to have only minimal impact on individual performance within the broad ranges of the categories which have been measured.
- Therefore, tools and aids to improve productivity in the systems development process are dependent on the lower levels of the productivity pyramid. Even given the bottom three levels as a solid foundation, it is apparent that the quality of the personnel assigned to a given project will determine not only the ultimate success of the project but also must be considered in selecting the appropriate tools for use by the development team. There is a vast difference between individuals



who "think" in terms of card image records, relational tables, and four dimensional arrays, and what is user-friendly to one person is going to be either incomprehensible or of little use to another.

- Given the necessary priorities of the productivity pyramid, there still remain no "cookbook" solutions to the continuing productivity problem. As Gopal Kapur, an experienced observer and practitioner of productivity improvement, has stated: "It is possible for two people to shop at the same store, buy the same ingredients, use the same stove and utensils, and cook the same recipe, but there is always going to be a big difference between a short order cook and a world class chef." With that bit of wisdom, let's analyze the productivity problem in more detail.

B. PROBLEMS VERSUS SOLUTIONS

- Fundamentally, the problem has remained the same--instructing computers what to do is a detailed, tricky business which is highly error prone. This means that developing computer applications has been a relatively slow process. On the other hand, the power and promise of the hardware technology keeps increasing rapidly, and this results in increased expectations on the part of end users. This has been especially frustrating for:
 - Vendors who want to install new hardware as rapidly as possible.
 - End users and executives who have been sold (by vendors and consultants) on the wonders of computers only to find that it takes an inordinate amount of time for their central DP, EDP, ADP, MIS, or IS departments to get anything done for them. Then, when they do get something done, it costs too much both for development and operation.



- Over time, the problem has become compounded by the fact that maintenance (and even conversion) of existing systems takes priority over the development of new systems. Estimates of maintenance costs over the systems life cycle normally run at about 50-60%, and there are many who consider these estimates to be low.
- Languages (of all generations), DBMSs, operating systems, report generators, interactive computing (timesharing), and now personal computers were all supposed to eliminate the barriers to computer access and facilitate the development of applications. However, the problem has not gone away; except for the most trivial of applications, the problem of slow and costly systems development is still with us. One of the primary reasons mentioned for this state of affairs is that systems are becoming more complex.

1. SYSTEMS COMPLEXITY

- One of the organizations which have been interviewed during the course of INPUT's research on software productivity is a major railroad which served as a test site for early punch card equipment, making it literally IBM's first customer. The vice president of MIS joined the railroad out of college over 25 years ago and has been with them ever since. During his tenure he has:
 - Developed compilers and helped install major IBM computer systems without using any IBM software.
 - Assisted in the conversion effort necessary when his company became one of the first all-COBOL shops in the world.
 - Participated in the Data Base Task Group within Guide before IBM had any data base products. (The task group was prompted by GE's development of Integrated Data Store.)



- Mandated "paperless" programming (by eliminating all card decks and program listings) at a time when most companies were considering whether programmers should have terminals.
 - Successfully installed an early IBM Mass Storage System (3850) for on-line tracking of car movements.
 - Became something of a pioneer in distributed processing by installing 350 minicomputers for waybill preparation and distribution.
 - Integrated an operations research group into the MIS department to address operating problems.
- When asked about the increased complexity of the systems, he stated that the fundamental applications had remained essentially the same over the years. The complexity came in the hardware/software systems environment which had evolved over the years. He said he could not understand why it took so long to get certain applications developed, but he did know that fair comparisons could not be drawn between the freewheeling early days and the structured environment he had helped to create. He made the following specific points:
 - There were some applications which had essentially the same government-required output which required more maintenance effort per year than they had originally required to develop 25 years ago.
 - Given a relatively straightforward application of 25 years ago, with the requirement to produce the same outputs and serve the same essential functions, it would now require from two to five times the amount of development effort. There are a number of factors which contribute to this:



- The procedural aspects of project authorization, initiation, and control add front-end expenses which must be considered.
- The organization has more layers. Several levels of management will be involved in any project of significance.
- Depending on project size, it may require the use of a design methodology which usually means an additional person will be involved from the beginning.
- There are specialists for analysis, design, and implementation, and this means several people are involved and interacting; there are no one-man projects.
- Practically all projects must be integrated into the DBMS and with other applications systems. ("As soon as you collect any new data someone wants to use it, or you must arrange to obtain some of it from another system.")
- There is a tendency to make all systems interactive, that is what end users expect and what systems developers like to work on, so even systems which should be batch wind up on the network. (A specialist in telecommunications systems may be required.)
- It appears that it is not so much a question of the applications becoming more complex but the systems development environment becoming more complicated. More people have become involved which makes interpersonal and interorganizational communications more difficult, and the choice may be made to make the systems more complex and responsive than they need to be. The result is added cost for essentially the same information.



- When the interviewee was informed that Business Week had stated his company was using artificial intelligence, his response was that he guessed they might be referring to a dispatching system they had implemented. Disavowing that he understood what people meant by artificial intelligence, he simply stated: "As far as I'm concerned it is straight linear programming, and it is used primarily for training new dispatchers, but if they want to call it artificial intelligence I guess that is O.K." The only problem is that pretty soon someone will want to make the dispatching system a "real expert" system and a LISP machine will be installed, and then all the scheduling systems on the railroad will have to be integrated with the new dispatching system even if it is not used to actually dispatch trains.

2. MATCHING PROBLEMS TO SOLUTIONS

- A few years ago, during research on productivity, a vice president of the SAS Institute was interviewed who maintained that there was not a productivity problem. The only problem as he saw it was that we were all working on the wrong problems. During the course of the interview, it became apparent that what he meant was that we were spending an abnormal amount of the total IS resource fitting problems to solutions rather than working on the problems themselves. He specifically cited systems programmers (he was one) and "PC jockeys" as being part of the problem rather than the solution. At first, this seemed to be a rather contrary and contentious attitude, but the more we thought about it the more truth there seemed to be in the statement.
- If one totals up the time which has been spent converting applications to different languages, operating systems, network architectures, methodologies, data access methods, and DBMSs, it is really a wonder that the IS department has been as responsive to end-user requests as it has. Perhaps it is true that we have been working on the wrong problems by chasing the Eldorado which is always just over the horizon. There can be no question that an enormous amount of resources has been expended to no practical effect except to keep up with the latest hardware/software technology.



C. SOLUTIONS VERSUS PROBLEMS

- While fitting problems to solutions may be considered working on the wrong problems, the solutions themselves can be distracting enough without ever thinking about the problems which are to be solved. At the time of the 1963 IBM survey, the language choices for IBM customers were relatively simple—Assembly Language, Fortran (for scientific), or COBOL (for commercial). The conclusion that programmers did not like to change languages was presented to IBM management at the highest levels in the company. The result was that IBM decided it would "simplify" the decisionmaking process by announcing one language (PL/I) for its System/360 line of processors. This "solution" did not solve anyone's problems. IBM got stuck with another language to develop and maintain, and users had yet another choice to make.
- The situation in the last 20 years has only gotten more complicated. Fortran and COBOL are still around, IBM still likes PL/I, Basic and Pascal are widely used, C and Ada are upon us, LISP and Prolog have emerged from hibernation in the universities, and the proliferation of FGLs (fourth, fifth, and future generation languages) is practically unlimited.
- Now IBM also supports various operating systems, access methods, DBMSs, teleprocessing monitors, and networking schemes—a virtual supermarket of solutions. And down at the PC level, the choices are literally unlimited.
- All systems software addresses problems of productivity by making computers easier to use, but the choices for large users have increased enormously. Standards which were painstakingly established years ago (operating systems, languages, DBMS) are now threatened by the increasing number of products which are becoming available. Productivity solutions have always changed more rapidly than the underlying productivity problem, but selection of the appropriate tools (or even direction) is becoming an increasing challenge for computer users at all levels from PCs to 3090-400s.



I. MATCHING SOLUTIONS TO PROBLEMS

- Years ago, it was possible to bring in a simple report generator and have a couple of people try it out. It ran against existing "files" and could be easily tested. It was possible to get some feel for whether it would solve certain problems within the organization. Then if it was installed for general use and attracted enough users, it might eventually be made a standard within the installation.
- Then, with the early 4GLs, it rapidly became apparent that a fundamental change would have to take place in file organization in order to accommodate inverted file structures. Then, it became necessary to make a decision to try it on a project with the understanding that some investment and risk was involved. The following is an early case study of a pilot project for a well known 4GL over 10 years ago (before the term 4GL was used):
 - An enterprising analyst/programmer (more analyst than programmer) brought in the 4GL on a trial basis to try it for his end user--the personnel department.
 - He brought up some sample files and demonstrated the capability for terminal inquiry and ad hoc reporting. This was demonstrated at the highest levels within the company.
 - The test was deemed a success, and the necessary file conversion was started. It was then discovered that the "system" made no provision for updating the base personnel file; this would have to be done in COBOL.
 - All of the necessary interfacing with other systems had been ignored--Did the personnel system drive the payroll system or did the payroll system drive the personnel system? How secure was the terminal



query system? What paper procedures were required to drive the revised system?

- Developing the type of personnel system the company really wanted was far beyond the capability of either the analyst/programmer or the 4GL. In addition, the cost of developing the system was out of all proportion to the test of the 4GL.
- The analyst/programmer picked up the glory and the IS department picked up the pieces and enhanced its reputation for being unresponsive. Many IS departments remember similar experiences, and it dominates their reaction to this day.
- Whether the lesson was learned with a 4GL or a home-grown query and reporting system, it became apparent in the late 1960s and early 1970s that a management information system without supporting data was of little practical use. The ability to evaluate a DBMS is a problem which goes far beyond that required for a 4GL. More commitment is required in terms of the initial cost and in the residual costs of operating the system. Anyone who can remember the days when someone would casually suggest: "Why don't you bring in IMS and try it?" knows that bringing in a DBMS requires not only courage but a certain amount of faith. The public argument concerning the relative merits of the relational model (or even what it is) does not help users very much, and not only are they confronted with IBM's dual DBMS strategy, but data base machines are becoming viable alternatives.
- As the more comprehensive integrated applications development systems appear, the decisions become even more difficult. Fair tests require commitments on major projects; the term "vaporware" has come into being and is enough to make many IS departments very nervous. Improper applications of productivity tools have resulted in major systems failures which have been publicly reported. Fitting solutions to problems is a far from trivial exercise. Major shifts in the tools and aids used for systems development put



at risk not only careers but the performance of the organizations which try them.

2. SOLUTIONS BECOMING PROBLEMS

- Evaluating, much less testing, of various productivity improvement alternatives can become a significant part of the general IS overhead. Once the decision has been reached to test or install a particular applications development system, it is difficult to find personnel who are experienced in its use so the cost of training must be added to the already significant expense of the product itself. And, in any large organization, there will be the dissenters just waiting for the project to fail. All of the talk about applications programmers (and COBOL) disappearing in a certain number of years does not help in establishing a productive environment during the period when everyone is waiting for the miracle.
- In addition, many productivity tools and aids produce so much paper that the management of documentation becomes a problem in its own right. The logic of what the system does becomes lost in the boiler plate of the methodology and its associated diagrams and data definitions. Producing these paper documents seems to become an end in itself, and the programs are practically incidental.
- It also seems that most of our productivity tools have taken us in two directions:
 - One is toward specialization of parts of the systems development function. The development center, the information center, and those using PCs all use different sets of tools, but eventually their systems must come together. It makes the hardware connectivity problem look simple.



- The other is to aim tools at lower and lower levels in the aptitude range. The anthropomorphism associated with computers from the beginning persists in the desire to have them understand natural language. It is doubtful that giving someone on the low end of the aptitude range the ability to talk to a computer is going to have any practical results. It is difficult to explain complex procedures in natural language; that is the reason scientific notation was invented in the first place. In fact, Dr. Edsger W. Dijkstra has decried computer anthropomorphism precisely because it disguises the computers greatest strengths. These strengths are in complex computation and logic and not in printing relatively simple reports from established data bases.
- It is INPUT's belief that both of the trends can be counterproductive in the systems development process. In fact, the communications problems associated with increased specialization and developing tools suited for lower levels of the performance range is one reason the productivity problem persists.
- The difficulty in improving productivity in the systems development process is a subset of the overall problem of improving productivity in the office. This problem seems to stem from the fact that offices are not factories, and the techniques employed successfully in factories do not work in many office situations.

D. FACTORY VERSUS OFFICE

- In the famous Hawthorne Experiment, industrial engineers determined that a group of workers demonstrated improved productivity whenever their working environment was changed whether the changes were "good" or "bad." The reason was discovered to be that the workers were responding to being singled



out for attention, and they were responding in a positive manner regardless of the specific change which was made in their working environment. It is probable that office workers respond in a similar fashion, but the impact of changes are substantially more difficult to control and measure.

I. THE MEASUREMENT PROBLEM

- To appreciate the difficulty of measuring office productivity, it is necessary only to review some rough breakdowns of how office workers spend their time (see Exhibit III-4). Occupational categories A and B (Executive, Managerial, Administrative, and Professional and Technical) constitute over 68% of all office expense (not shown on the chart), but they spend their time on activities which defy accurate measurement by industrial engineering standards. (This chart first appeared in Impact of Office Systems on Productivity, INPUT, 1983.) Everything depends on the quality of the decisions or analysis which is made and not on the activity itself.
 - Speeding report preparation does not necessarily improve the contents of the report.
 - Evaluating the value of telephone conversations and interpersonal communications is impossible; different people have different styles for obtaining information.
 - An observer (or time and motion engineer) cannot determine whether a knowledge worker staring into space is deeply involved in analysis or decisionmaking or mere thinking about his golf scores. Counting "therbligs" does not work well for office workers.
 - Categories 4 and 5 were intentionally minimized in the report because such use was determined to be casual. With more managerial and technical employees using personal computers, both could become substantially more important, but it remains difficult to determine



EXHIBIT III-4

DISTRIBUTION OF OFFICE WORKERS' TIME

OCCUPATIONAL CATEGORY FUNCTION	(A) EXECUTIVE, MANAGERIAL, ADMINISTRATIVE	(B) PROFESSIONAL & TECHNICAL	(C) SALES WORKERS	(D) SECRETARIES, ADMINISTRATIVE ASSISTANTS	(E) TYPISTS, DATA ENTRY	(F) CLERICAL
① Analysis & Decision-making	15.0%	35.0%	7.5%	12.0%	5.0%	25.0%
② Report Preparation	22.5	20.0	12.5	7.0	5.0	5.0
③ Dictation	0.5	0.0	0.0	5.0	0.0	0.0
④ Typing/Data Entry	0.0	0.0	0.0	20.0	40.0	10.0
⑤ Copying/Information Entry	0.0	0.0	0.0	6.0	12.0	12.0
⑥ Information Handling/Storage	12.0	15.0	5.0	15.0	15.0	30.0
⑦ Telephone	20.0	10.0	25.0	17.5	10.0	8.0
⑧ Interpersonal Communications	30.0	20.0	50.0	17.5	13.0	10.0
Total	100.0%	100.0%	100.0%	100.0%	100.0%	100.0%



whether the time is productively spent. In the final analysis, you would have to assume that a \$100,000 a year manager keying data into a spreadsheet has decided it is productive use of his time. (These questions of office productivity will be explored in a new series of INPUT reports in 1987--Distributed and Office Systems Directions--which will complement and supplement our current Large-Scale Systems Directions report series.)

- Systems development personnel fall within these two occupational categories which are so difficult to measure. Adequate measures of programmer or applications development productivity do not currently exist. Whether it is lines of code, function points, meeting schedules and budgets, or some of the more comprehensive software engineering models, all suffer from serious deficiencies in terms of measuring the productivity of either the individual or the project team.

2. PERFORMANCE MEASUREMENT DEFINED

a. Blue Collar Workers

- Factories produce physical objects which can be counted with quality which can be inspected or tested. Statistics provide highly satisfactory mathematical models for purposes of quality assurance. Accounting provides adequate means to provide general productivity measurement in terms of unit costs and distribution of these costs.
- It has been found that work simplification on the assembly line generally increases individual productivity in terms of contribution to the finished product. The assembly line moves faster when each workstation performs a relatively simple task. Where the tasks vary along the line, additional workers will normally contribute in a linear manner. (For example, if task A can be performed at the rate of 10 per hour by one worker, two workers will produce 20 per hour.)



- Once programmed (laid out) for human beings, an assembly line is relatively inflexible and there is little opportunity for individual contribution or individuality on a day-to-day basis. Properly programmed robots will assist materially in making the assembly line and manufacturing process less costly, and perhaps more flexible.

b. White Collar Workers

- Offices deal with data, information, and knowledge and produce information, usually in paper form, but increasingly on computer display screens. Individual contributions are most frequently verbal, but most of significance are reduced to paper. The value of the information produced is generally unrelated to its volume (the number of pages produced) or its cosmetic qualities. Indeed, fancy reports may tend to obscure the fact that the information content is inaccurate and misleading, and worthwhile information can be obscured in a sea of boilerplate, jargon, and buzzwords.
- INPUT has determined that the productivity of the office must be measured at four levels:
 - The hardware/software level.
 - The human/machine dyad.
 - The work unit level.
 - The institutional level.
- These levels are not necessarily directly related. For example, the emphasis we place on "user friendliness" and anthropomorphism at the human/machine dyad places an extreme burden on performance at the hardware/software level. While computer (and software) vendors like us to believe that machines



are cheap and people are expensive, there is obviously a point where this logic breaks down. We have learned that it is possible to develop systems with such poor hardware/software performance that they are impractical. This will always be true regardless of improvements in computer hardware/software technology. For example, it is possible to prove that certain algorithms of operations research and artificial intelligence are transcomputable, which is to say that it is physically impossible to build computers which can perform the calculations necessary for the "solution."

- Similarly, it is possible to state that maximizing work unit performance may impact the performance of the individual. Having a worker ten times more capable than other members of a team has adverse impact on the outstanding individual's performance. (During World War II, early efforts in operations research indicated it was advisable to have high-speed troop transports travel out of convoy without protection rather than be limited by convoy speeds; this is true on many systems projects.) It is also possible to have high productivity at the first three levels and have poor institutional performance (remember, the chef is important).
- The information systems function, being a subset of the office environment, must have its performance (productivity) measured at all four levels. However, the information systems function has assumed a central role in the development, production, and quality control of the actual office "product"--information. It is, therefore, important for the IS function to understand data, information, and knowledge so that its responsibility can be defined.

E. DATA/INFORMATION/KNOWLEDGE QUALITY

- Terminology in the computer industry has progressed much more rapidly than have the underlying hardware/software systems which have been developed. While we have proceeded from "data processing" to "information manage-

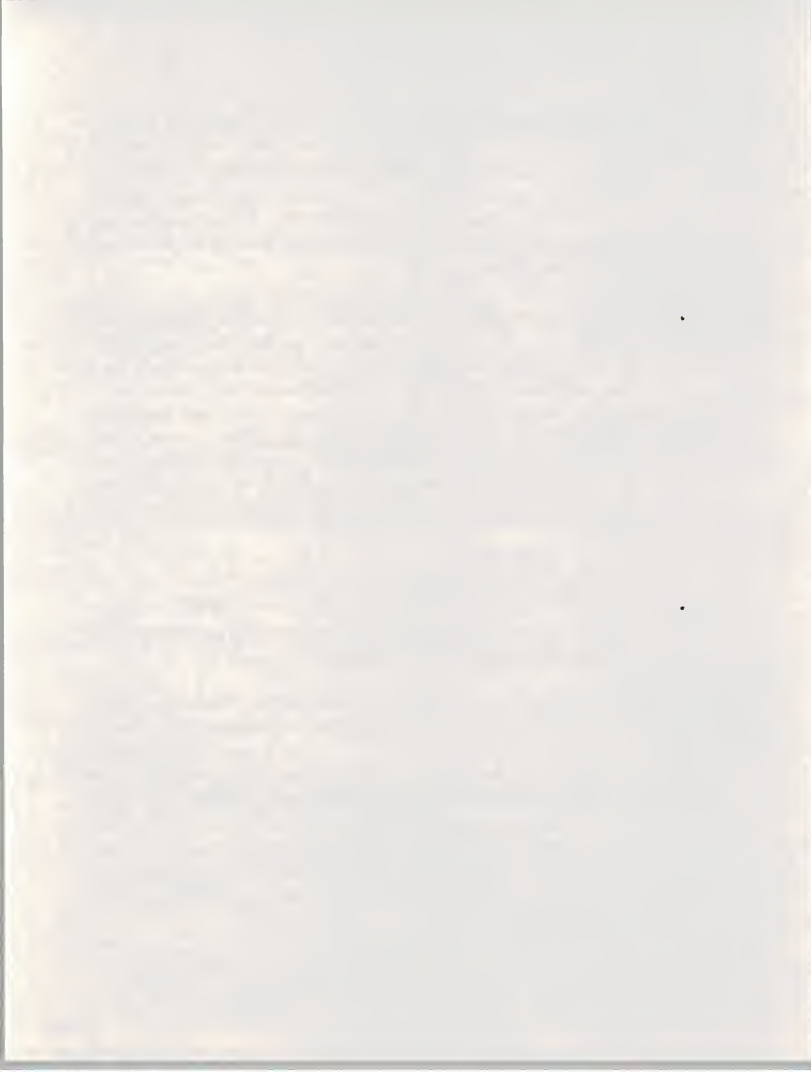


ment" to "knowledge engineering," major computer applications have not really changed that much, at least those coming from many development centers. This is more than a problem of low productivity. There are fundamental differences between what we are saying and what we are prepared to do or are willing to accept full responsibility for. Expectations, and even promises, of the 1950s are still beyond the horizon of our capabilities. Everything is always over the next hill.

- For example, it has been recently reported by CompuServe that Government Computer News stated that Paul Strassman, a retired Xerox executive, has said that as banks have applied more and more computer technology productivity has actually declined. In fact, it was reported that he stated in an eight-year study on what makes a company successful that information technology is not even in the top 10 of the most significant success factors (this being an example of institutional performance measurement). This particular information is given not only to support the information in the preceding paragraph but also to be played against the definitions which follow.

I. PRACTICAL DEFINITIONS

- In INPUT's recent study on user applications of CD ROM, it was found necessary to present some practical definitions of data, information, and knowledge. Those definitions are as follows:
 - The definition of data comes from the late Fritz Machlup, who finally concluded, after reviewing how far data had strayed from its original Latin definition of "the givens," that: "This semantic muddle, however, need not cause any serious trouble because the arguments in which data, whatever they are, play a central role are relatively simple--data entry, data storage, data retrieval, data processing, data services, and all the rest, refer simply to things fed into a computer. These things, now data from the point of view of the programmers, operators, and users of the computer, need not be data in any other sense" (his emphasis).



- Information and knowledge have a firm link, and the best way to define them is by distinguishing between them. The commonly accepted distinctions are as follows: (1) information is piecemeal, fragmented, particular, whereas knowledge is structured, coherent, and often universal; (2) information is timely, transitory, perhaps even ephemeral, whereas knowledge is of enduring significance; and (3) information is a flow of messages, whereas knowledge is a stock, largely resulting from the flow, in the sense that the "input" of information may affect the stock of knowledge by adding to it, restructuring it, or changing it in any way (though, conceivably, information may leave knowledge unchanged). An additional fundamental distinction is that information is acquired by being told, whereas knowledge can be acquired by thinking (without new information being received).
- Using this definition, it is possible to decide that CompuServe is a data processing services company which provides information services. In the particular case cited above, information was taken from another source and processed so as to make it more widely and rapidly available. This information makes no pretense as to quality, much less as to whether this information alters the generally accepted knowledge that computers improve productivity. That is for you to decide, and you must consider the following:
 - Who is Paul Strassman? Does it help to know that he was a vice president at Xerox and was active in the American Management Association?
 - Were his conclusions accurately reported in Government Computer News? Did the reporter for CompuServe correctly interpret what was read in Government Computer News?
 - Are the data on CompuServe secure or has some disgruntled hacker decided to edit them at random?



- Who is writing this report for INPUT? Was Paul Strassman called to ask what he really said and qualify the research which went into his study?
- Is all of this just an idle exercise in semantics? We think not. It is important that IS professionals understand the responsibility they have for data/information/knowledge quality within their organizations. And, oh, yes, it was also reported that "key executives" in the banking industry stated they were going to "reduce the rate of increase" of computer budgets because they cannot detect productivity gains from automation.

2. BEYOND SEMANTICS

- After 30 years of computer systems development, it is possible to draw some general conclusions about data, information, and knowledge:
 - Data, by definition, are stored in computer systems. However, it is possible to be more specific than that; data of institutional significance remain on host mainframes (for any but the smallest organizations), departmental processors are used primarily to concentrate data for specific work units, and personal computers are used to generate paper documents (correspondence, reports, etc.).
 - Information is transferred by voice (being told) or by paper documents. While the transfer of information by voice represents a substantially higher percentage of total office costs, the official communication of information remains on paper. If information of significance is generated in meetings or telephone conversations, it must normally be documented for purposes of validation, distribution, and storage. Paper remains the primary information media of organized human activity (business).



- Human minds remain the primary processors of information, and the brain remains the primary storage device of knowledge (a very small percentage of individual human knowledge is ever documented). The best research efforts in artificial intelligence have resulted in precious little knowledge as to how either the mind or the brain works (except the grudging admission that man did not create computers in his own image).
- Despite changes in terminology, it is a fair statement to say that the central development function is concerned primarily, if not exclusively, with data. Very few computer systems personnel have any interest in the vast flow of paper information within their organizations. If it cannot be brought on-line, it is ignored. Therefore, top-down design extends only to the printer. All of the problems of paper procedures--movement, filing, archiving, and disposal--are left to the "users," who as human beings are also the storage media for knowledge.
- The self-imposed isolation of most DP/IS organizations from the main information flow of the organization and from its knowledge bases (end users) has resulted in many systems which do not meet the needs of the organization. Paper-based systems and the essential organizational knowledge bases have not even been considered as "peripherals" to the computer/communications systems which have been developed. The objective of most "development centers" has been to ignore information flow and knowledge. When one looks at what a so-called "knowledge engineer" is doing in building an expert system, it inspires one to ask: "Isn't that what systems analysts are supposed to be doing?"
- On the other hand, the automation of offices has meant improving the production of paper documents with little regard for the quality of the contents. There is so much information flowing through offices that few office workers have time to devote to analysis; they are too busy generating pretty reports from their personal computers.



IV THE SOLUTIONS EVALUATED







IV THE SOLUTIONS EVALUATED

A. CONVENTIONAL APPROACHES

- Simply put, any computer application or system has five functions:
 - Input.
 - Processing.
 - Storage.
 - Communication (data movement/transmission).
 - Output.
- In a relatively short period of time, technology has progressed to the point where a single large-scale processor can perform more calculations than all living humans, entire paper filing systems and even libraries can be kept on-line (optical memories), and communications links have capacities which exceed even the computer's ability to process the information. The primary limitations on using this enormous potential are input and output.
 - On the input side, information must be translated (encoded) from human representation into that of the computer. This information essentially consists of the following:



- Instructions to direct the processing of the computer.
 - Structure for data representation and storage.
 - Data of all kinds (numbers, characters, text, bit patterns, graphics, pictures, etc.).
- On the output side, data must be translated (decoded) into some form of human representation. These representations are fundamentally:
 - Paper information.
 - Electronic information display.
 - Direct action (process control, robotics, audio response, etc.) which would normally have been performed by a human being.
- Exhibit IV-1 presents a schematic of the primary ways information is translated for computer use (input) and retranslated into information for human use and/or into direct humanlike action (output). This schematic will be used to identify and evaluate the various solutions to the human/machine information exchange problem.

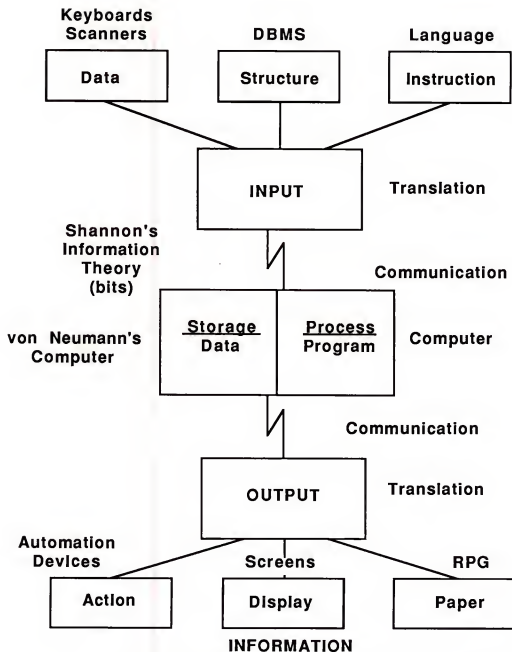
1. LANGUAGES

- The original and primary tools for giving instructions to computers fall under the general classification of languages. From simple symbols for machine operation codes, languages have evolved in many directions, and only two things are certain at this time--languages are proliferating despite all efforts at standardization, and they remain the most debated subject in the programming community. During the course of the research for this study, we called an old-timer who has been involved with languages a little too long and he passed along the following "jokes":



EXHIBIT IV-1

A SCHEMATIC FOR EVALUATION





- Did you hear about the LISP programmer's son? He can tell you all about his grandmother, but he can't add 2 and 2.
- Greek looks like APL to me.
- Read any good COBOL programs in the last 20 years?
- PL/I isn't, is it?
- 4GLs are great. I think they may replace 407 board wiring.
- Is kanji a natural language?
- Like everyone else, INPUT has some very definite opinions about languages. They are as follows:
 - There is no one best language. The appropriate language depends on both the application and individual who will be using the language.
 - The concept of language generations (especially the 4GL) is fuzzy at best and down right misleading in normal use. It is our opinion that languages evolve along general paths such as symbolic, algebraic, English, query (information retrieval), report generation, etc., and FGL (first, fourth, fifth, and future) is as meaningful as 4GL as it is currently being applied.
 - Natural language is not appropriate for instructing computers in what to do, and the quest for talking to computers is, in most cases, misguided and a waste of effort.
- Enough of that, it makes us feel the same way we did in the early days of COBOL when it was suggested that executives could "read" COBOL programs



and find out what the programmers were doing. This in turn reminds us of the advertising copy for many of the solutions which are currently being proposed to solve the productivity problem in the systems development process. Let's look at some facts.

- Exhibit IV-2 shows the primary languages which are currently in use among the respondents to this study. The following seems to be apparent:
 - Those systems managers who vowed they would never leave Autocoder for COBOL in 1963 have either changed their minds or been replaced because COBOL is the primary language of 63% of the respondents.
 - Fortran remains the primary language of 17% of the respondents, even though it was the earliest "higher level" language.
 - With all of the might of IBM behind it, PL/I has not attracted very much of a following after over 20 years.
 - Among the "other" primary languages from our sample are: Assembler, "C," Focus, Mantis, and Model 204.
- When asked about the use of 4GLs, we got a mixed bag of responses (see Exhibit IV-3).
 - Focus was the most popular, with 21% of those having one installed; SAS, Ideal, Nomad, and ADS/On Line received a couple of mentions; and several respondents had several such tools installed and it was obvious that most of the development managers interviewed definitely considered them to be peripheral to the central systems effort.



EXHIBIT IV-2

PRIMARY LANGUAGES USED

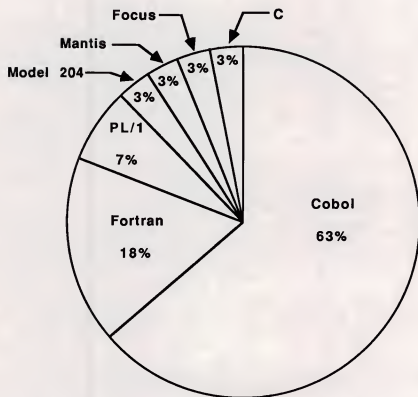
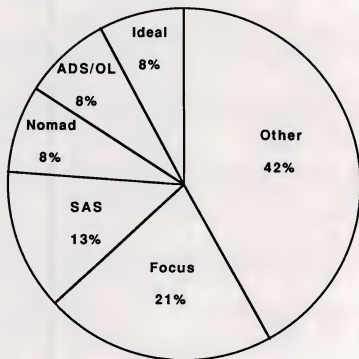




EXHIBIT IV-3

FOURTH GENERATION LANGUAGES INSTALLED





- Among the "other" category was a great variety of responses including PL/I, DMS, SPSS, IFPS, Intellect, Datatrieve, and a couple of home-grown packages.
- Perhaps even more striking was the fact that several of the development managers interviewed thought their companies had a 4GL installed but they were not sure of the name. We take this to mean that these managers do not take all of the talk about 4GLs seriously enough to even consider them on their projects.
- Products generally classified as 4GLs have been reported to improve productivity on specific projects by as much as four or five times. (Projects implemented in 4GLs take one-fourth or one-fifth the time to implement as would the same project in COBOL.) These claims have been confirmed in specific tests made by a number of end-user organizations. The primary question seems to be what percentage of applications lend themselves to implementation with 4GLs (or what percentage of major projects can be implemented using 4GLs).
- Since interviews were conducted with systems development personnel (as opposed to those in an information center), two questions were asked concerning the relative effectiveness of 4GLs in a project environment.
- Respondents were rating various tools and aids on the customary 1 to 5 scale where 1 = not important and 5 = very important.
- Then respondents were asked another question: "If you could not use the specific tool or aid on a particular project, how much would development costs increase?" It was felt that this question would make development managers think about actual situations and place a value on the tool or aid independent of popularly reported test results. It was also designed to get some feel for what percentage of major projects might lend themselves to 4GLs.



- The mean rating of 4GLs on the importance scale was 3.12 which is considered to have very little significance because INPUT has found from long experience that ratings on a 1 to 5 scale tend to cluster between 3.0 and 3.9, and it is only those ratings which fall out of that range which indicate positive or negative reactions on the part of the sample population. In other words, the systems managers surveyed do not consider 4GLs of great importance in improving productivity in systems development nor do they consider them to be insignificant as tools.

- That type of analysis is nice and simple, and INPUT has had long experience with using 1 to 5 ratings to isolate factors of real importance. However, the second question adds several other dimensions to the relatively neutral rating given to 4GLs. The mean percentage of increase in development costs which would be required if 4GLs could not be used on development projects was 58%, but then the analysis becomes more complicated.
 - This overall estimate of the increased cost which would result in not using 4GLs would indicate that savings of 37% could be realized on the average development project which is not currently using 4GLs. While this is not insignificant, it is obviously considerably below the 75-80% savings which would be represented by the published claims of 4GL vendors.

 - In fact, if the responses for primary languages other than COBOL are eliminated from the results (along with two exceptionally high responses which will be detailed later), the savings being realized from use of 4GLs in COBOL shops would appear to be 26%. Past INPUT research has indicated that approximately 25% clearly demonstrable cost savings is necessary to get users to consider what they perceive to be a major hardware/software systems change. Since considerable investment in software and training is necessary in order to install and make effective use of 4GLs, the perceived savings of development



managers tends to fall well within a comfortable range of inertia (less than 25%).

- This means that approximately 35-40% of the work in the COBOL shops would have to be considered appropriate for 4GLs in order to achieve the 26% cost savings which was mentioned above. It would not appear that COBOL development managers feel 4GLs are ready to replace all commercial applications development. However, the other development environments also provided some significant insights into the relative effectiveness of 4GLs as productivity tools. Consider the following specific examples:
 - First, there is the installation which classified PL/I as a 4GL, and here the respondent estimated that he would increase project cost estimates by 100% if he could not use his "4GL." (Presumably this is compared to assembly language, and the estimate is very much in line with the productivity increase which was experienced when going from Autocoder to COBOL.)
 - The installation which listed Focus as the primary language also stated that a 100% increase in cost would result if it could not use that language. (Presumably the work not done in Focus would be done in COBOL, and this would mean that approximately 70% would be done in the 4GL and 30% in COBOL in order to achieve 50% savings.)
 - A 100% increase in cost was also predicted for a shop which stated it was "multilingual" with PL/I as the "third generation language" and Info as the 4GL.
 - Three Fortran shops predicted 25%, 50%, and 75% increases if they could not use their 4GLs--SAS, Datatrieve, and Focus, respectively.

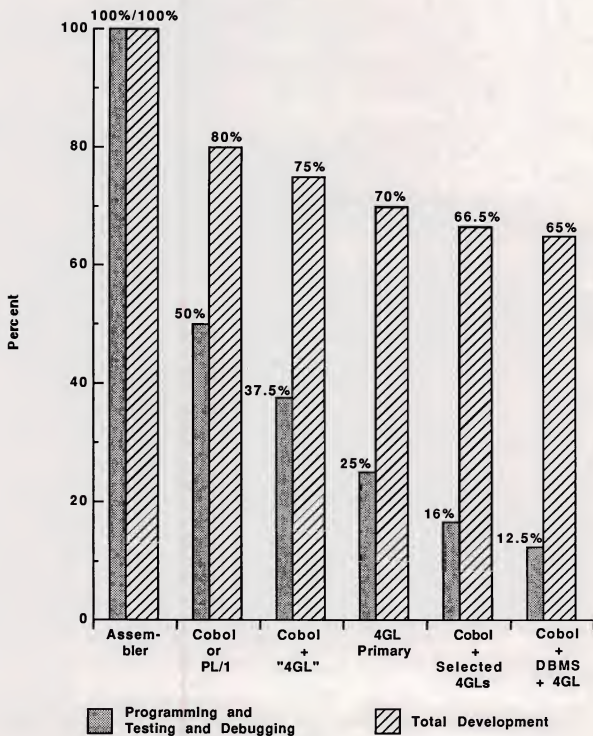


- Where Mantis was listed as the primary language, the respondent stated costs would only increase by 5% if Ideal could not be used for some of the work.
- Where Model 204 was listed as the primary language, it was stated that costs would only increase by 10% if Nomad II could not be used as a 4GL.
- Now we return to the two shops which listed COBOL as the primary language and predicted development costs would rise by 200% and 300%, respectively, if they could not use their 4GLs. (These were the two which were excluded when the 4GL "savings" for COBOL shops was determined to be 26% above.) The respondent which estimated 200% increased costs without 4GLs was using SAS and Focus, and the one which estimated a 300% increase was using IDMS and ADS/Online (and he tied the two together in making his estimate).
- All of the numbers presented above appear to us to be quite reasonable. Essentially, they say that if we had to abandon our current languages and go back to our previous way of doing things it would indeed be costly. Exhibit IV-4 summarizes these approximate findings in terms of the perceived savings (value) of the 4GL tools being used by the respondents. The base is established as Autocoder (assembler) to show the progress which has been possible through use of higher level languages over the last 20 to 25 years. A few comments are in order concerning these savings:
 - Improved productivity as a result of languages result in savings which can be applied primarily to the programming and testing and debugging phases of the development process. Since these phases represent approximately 40% of systems development costs (see Exhibit III-1), the savings attributable to languages bottom out rather rapidly when applied to the overall development process. If the assumption is made



EXHIBIT IV-4

SAVINGS FROM HIGHER LEVEL LANGUAGES





that the respondents estimates were based on selection of a language to implement a defined and designed project (and we think that assumption is reasonable), the true savings may be substantially less than if we assume the estimates applied to the overall application development cycle.

- It is also apparent the big savings come from selecting a variety of tools (SAS and Focus), and that DBMSs form the foundation of the savings from 4GLs (IDMS and ADS/Online).
- The importance of both DBMS and design methodologies in the view of applications developers was clearly demonstrated by research results which will be discussed later. The important point is that languages can contribute only to telling a computer what to do--process in this fashion or produce this report. They are concerned primarily with performance at the human/machine dyad.

2. OTHER TOOLS, AIDS, AND METHODOLOGIES

- Languages are fun to develop and argue about; that is the reason we have so many. Unfortunately, it has been found that the most elegant and productive language is of little use if a thorough job of problem analysis and systems design is not done. Then, regardless of how well designed a system is or how good the implementation language is, the system will not be effective (productive) unless the data are available and of good quality. This was discovered after numerous "management information systems" were developed and failed because of inadequate data in terms of either availability or accessibility.
- Two parallel developments attempted to solve the problem:
 - Data structuring facilities in languages were enhanced. Information retrieval and reporting languages with inverted file structures had been



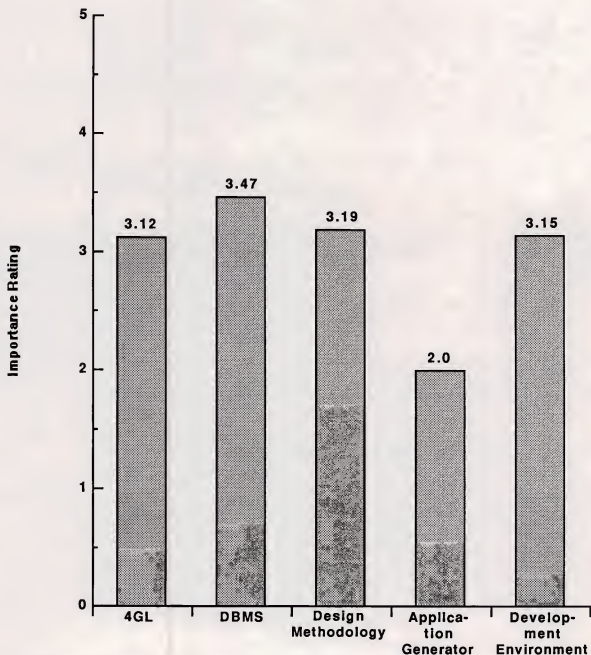
around for over 20 years before they were labeled "fourth generation languages."

- Concepts of shared data were developed in parallel with the development of direct access storage devices. Integrated file systems such as Integrated Data Store (IDS) from the General Electric Company were around long before the term DBMS was applied to them.
- It soon became apparent that new concepts for data structuring were necessary and that ways of thinking about data and the structure of data themselves were two different things. Languages for describing data and for linking logical views with physical structures were necessary. It was also found that many systems personnel, who would embrace languages with a passion, had little interest in discussing data schemes much less the data themselves. As an MIS director responded to an INPUT interview many years ago: "I don't even like to think about data models; the whole subject bores me."
- Thus, a new breed of DBMS specialists was born, and they proceeded to split off into various schools based on their particular orientation and the politics of jockeying for position which occurs in any new profession which has significant implications in the marketplace for computer products. The controversy continues to this day.
- It was also necessary to bring some formalization to the design and implementation process itself, and structured programming and various design methodologies were developed. Once again there are competing schools of thought about these developments as well.
- One thing is certain, all three developments are necessary for a truly productive environment, and the results of our research indicates that this is so. Exhibit IV-5 compares the 1 to 5 ratings respondents gave to 4GLs, DBMSs, design methodologies, applications generators, and development environments (work bench, integrated tools).



EXHIBIT IV-5

RATINGS OF IMPORTANCE OF PRODUCTIVITY TOOLS,
AIDS, AND METHODOLOGIES



Rating: 1 = Low Importance, 5 = High Importance



- It can readily be seen that all of the ratings fall within the "normal range" of 3.0 to 3.9 except for applications generators. There is little significance to the fact that DBMSs are rated at 3.47 and 4GLs are rated at 3.12, although it would be nice to think that it is an expression of the fact that DBMSs are more fundamental to productivity improvement than the language used.
- However, there is significance to the rating for applications generators. A rating of 2.0 on a scale of 1 to 5 falls out of the normal range and indicates that the respondents to this study do not feel that applications generators are very important in improving productivity in developing applications. Analysis of this rating indicates the following:
 - Only four of the respondents had applications generators installed (DMS, UFO, ADS/Online, and a COBOL generator of unspecified origin).
 - There seems to be the general tendency to confuse applications generators with code generators which have normally been appended to the development process itself.
 - It would appear that "applications generator" is not a good term to use for some of the more advanced products which include DBMS and higher level languages.
 - It would also appear that 4GL, despite its lack of definition, has become one of those magic terms, whereas applications generator has a negative, or at least misunderstood, connotation. (Vendors take note.)
- When asked the question concerning how much the cost of a specific project would increase if the tools could not be used, the responses were as follows:

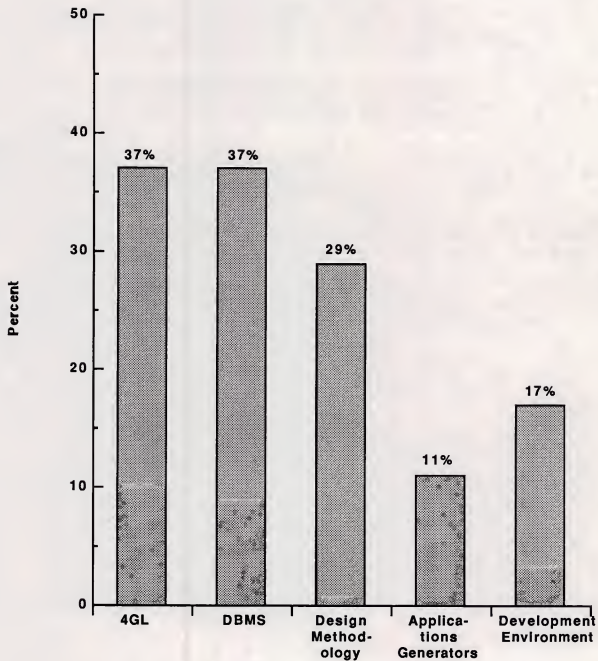


- As mentioned and analyzed previously, it was estimated that development costs would increase by 58% if 4GLs could not be employed.
 - The mean response for DBMSs was also 58% (actually 58.4% compared to 57.9% for 4GLs), and this rather remarkable consistency confirms the close identification of DBMS and languages with productivity improvement.
 - The mean response for design methodologies was 42%, for applications generators 12%, and for development environments 21%.
- Translating the increased costs if the tools could not be used into estimated savings which can be anticipated from employing the specific tools gives the results shown in Exhibit IV-6. Once again, on the surface these general estimates seem reasonable if we do not assume that they are cumulative. (Employing a DBMS, 4GL, and design methodology will obviously not result in negative development cost.)
- Unlike 4GLs, where IBM is not a significant factor, the most frequently mentioned DBMS was IMS (six respondents), and DB2, DL/I, and SQL/DS were among the systems which were installed. IDMS and ADRS were both mentioned by multiple users (4 and 3, respectively). The remainder of the installed systems were of great variety including IDS, the granddaddy of them all, installed on a Honeywell processor.
 - The design methodologies installed were predominately home-grown or so heavily tailored that they were considered their own. The only commercial products mentioned were SDM, AID, and Nomad II.
- The fundamental tools, aids, and methodologies of productivity improvement have been around for a long time. They address programming and data structuring within computer systems and the retrieval of information from



EXHIBIT IV-6

ESTIMATED SAVINGS ON
PRODUCTIVITY TOOLS





those systems (display and document preparation) (see Exhibit IV-1). There remains great controversy about the relative effectiveness of the tools. For example:

- Will 4GLs (whatever they are) replace COBOL?
- What is a relational data base system--what the inventor says it is or what DBMS implementors say it is?
- Then, there is "sub-second response time," a magical hardware-sensitive solution to the productivity problem made popular by hardware vendors. Here, even the basic research has been called into question in the public press (much less the conclusions which were reached).
- The fact remains that the true costs of developing systems have continued to increase as we have employed improved tools, aids, and methodologies. While part of this may be because systems are increasing in complexity, it is probable that there are other factors of equal or greater significance. First of all, let's analyze (roughly) the impact of the relatively crude tools which were available through the 1960s with those which became available during the 1970s. Exhibit IV-7 depicts such impacts for an IBM mainframe environment.
- The early tools to make computers easier to use were concerned with hardware/software performance as well as with performance at the human/machine dyad. There was much concern about such things as "compile speed" and "object program efficiency," and programmers were still concerned about how fast their programs ran. During the 1970s it became popular to accept at face value the premise that computers were cheap and people were expensive. This resulted in a severe hardware/software performance impact from the tools designed to make computers easier to use. Consider the following:



EXHIBIT IV-7

OPINIONS CONCERNING PERFORMANCE IMPROVEMENT
(1960s and 1970s)

PERFORMANCE LEVEL	IMPACT*	
	1960s	1970s
Hardware - Software	-1	-2
Human/Machine Dyad	+2	+1
Work Units	+1	+1
Institutional	+1	0

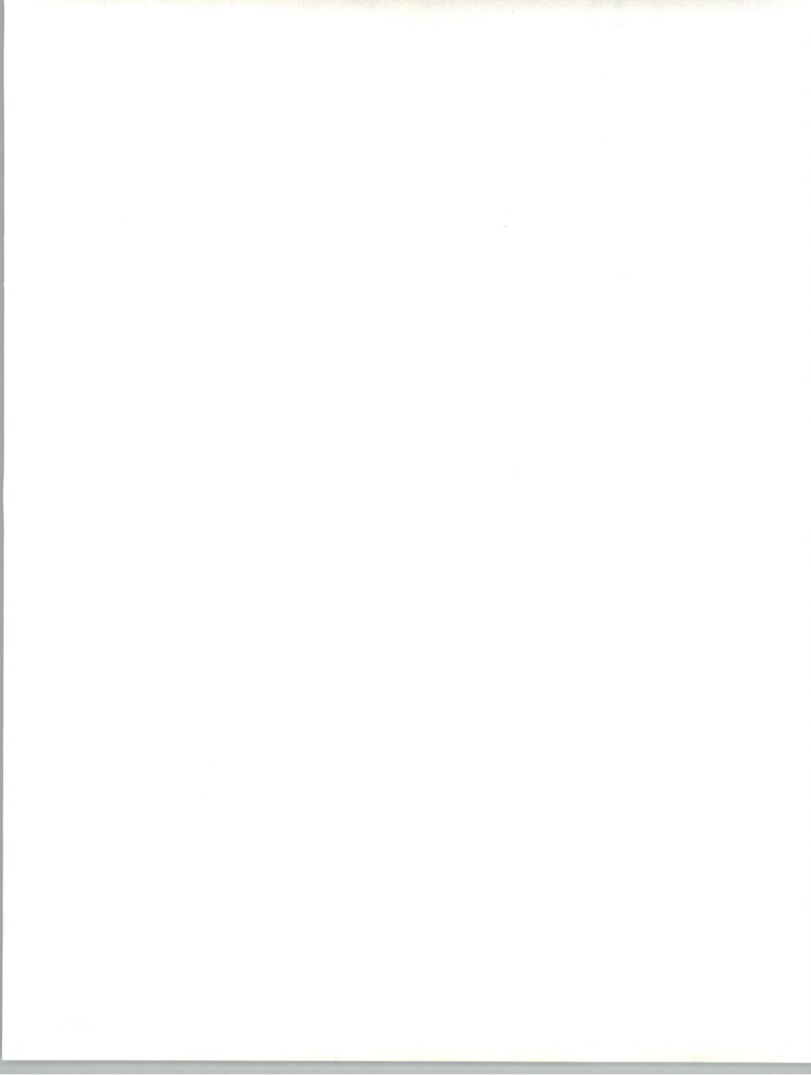
* -2 = Strong Negative Impact

-1 = Some Negative Impact

0 = Neutral

+1 = Some Positive Impact

+2 = Strong Positive Impact



- As operating systems evolved from DOS to OS to VS, the impact of problem program performance became relatively insignificant because it is lost under the burden of the operating system itself. A point of pride became that VS could keep CPU busy, but little emphasis was placed on what it was busy doing.
- The impact of DBMSs is nearly as dramatic; for example, major internal IBM systems which were developed in the 1960s could not be converted to IMS during the 1970s because the performance differences (in terms of transactions per second) would have required approximately six to seven times as much computer power to drive IMS, and IBM could not afford it.
- Then, of course, languages became decidedly less efficient in terms of machine usage--first in terms of compile time and object code and later in terms of the continuing burden of interpretation at execution time. (The mere fact that COBOL is now considered a performance standard would have been unthinkable in the early 1960s.)
- The benefits of early tools on productivity at the human/machine dyad were indeed dramatic. Not even the most dedicated computer freak (as they were called then) could argue that assemblers, Fortran, IOCSs, monitors, and early timesharing systems improved both personal productivity in programming but also accessibility to the computer resource. However, by the late 1970s (despite significant advances in the variety and even quality of productivity tools), the benefit was not so great. In fact, a new problem had become apparent:
- The tools themselves were beginning to add a new level of complexity to both programming and systems design. The operating systems had become so complex that as much time



was spent debugging JCL as was spent on debugging problem programs.

- DBMSs required a certain discipline and knowledge for the individual programmer or analyst in order to deal with both the logical and physical structures of data.
 - The complexity led to specialists in systems programming and data base administration who had to be consulted because individual programmers and analysts could not keep up with all the complexity of the operating environment.
- It was frequently the case that early computers were placed under the control of accounting departments because they were first used to automate accounting functions. There was favorable impact on accounting work units because the end users were intimately involved by having responsibility for the overall systems (hardware, software, and paper-based accounting systems). However, as the emphasis shifted from the early days of "saving money" to the more subtle emphasis of using computers to "make money," the control of the hardware/software became divorced from the work unit. The operating work units had to deal with some central facility for purposes of using the data processing facility. This resulted in the following:
- Despite dramatic increases in hardware (and even software), potential operating work units found they were dealing with an outside systems organization in order to get anything done, and the now infamous responsiveness issue developed.
 - The outside systems organization frequently remained under the control of the accounting (financial) organization of the company, and where it was recognized that this was not an appropriate place another lesson was learned--you can take an



accountant out of the accounting department but a bean counter remains a bean counter.

- Most systems organizations continued to remain oriented toward accounting and control as opposed to addressing the operating problems of their organizations, and the benefits of computer/communications technology were slow to reach the operating work units of the organization.
- The initial application of computers to commercial data processing problems usually resulted in tangible cost reduction benefits to the institution (enterprise). However, the efforts to use computers to "make money" had less tangible benefits to the institution. The benefits to the individual work units were frequently offset by the increased investment in computer/communications technology and the increasing cost of developing applications systems. In many organizations, bottom-line benefits became difficult to quantify. The ethereal concept of information as a corporate asset became prominent among vendors and prophets of the information age. On that note we entered the 1980s.

B. DISTRIBUTED SYSTEMS DEVELOPMENT

- Despite the advance of productivity tools during the 1970s, this was the decade during which dissatisfaction with the responsiveness of the central IS function resulted in the view that the central IS function was bottlenecked in applying all of the wonderful hardware/software technology which was becoming available. Management was still willing to spend money, and users were clamoring for new systems (or enhancements of existing systems). At the end of that decade, INPUT depicted the embattled IS function in a castle with users storming the walls clamoring for the treasures of the information



age. It is important to remember what really happened during the 1970s in order to put some of today's problems (and proposed solutions) into proper perspective.

- During the 1970s, the hardware and software systems to support "management information systems" were put into place. This was done at substantial investment in the following:
 - Conversion to virtual storage operating systems, which required substantial additional hardware investment in both processing power and main memory in order to run existing applications.
 - Conversion of existing applications to a DBMS environment. In IBM's case this was IMS, which rapidly got the reputation for using up more CPU cycles than even IBM could have imagined. (When asked about conversion costs to go to IMS, one respondent to an INPUT survey stated: "We don't know, and I don't think we want to know.")
 - It was discovered during this period that vendor-supplied hardware/software solutions do not, by themselves, solve user problems; data are required.
 - Users were encouraged to build enormous corporate data bases on the premise that data themselves have value, and some advocates of "information engineering" even suggested that the effort be capitalized and written off later since the expense could not possibly be supported by any current project.
 - Those who built the big corporate data bases frequently found that the problems had changed by the time the data bases were built, and then to their horror found that the very DBMSs they had installed (usually IMS) did not accommodate change very well.



- The big bang theory of systems development did not work very well, but a lot of data had been collected and part of the fallout was the conviction that it had value. Therefore, users stormed the walls shouting for results. Unfortunately, the IS function (as it was now called) had expended most of its development effort on building the data base(s) and not on the applications which used the data bases, and now it was discovered that just maintaining the large data bases and the major systems which used them was more than enough to keep them busy. In addition, dealing with very large data bases has turned out to add additional complexity to even relatively simple projects.

- So it was discovered that an enormous investment had been made over the decade in hardware, software, and data with very little visible improvement in institutional performance. Those responsible for all of the activities of the 1970s (vendors, management, and the IS function) were confronted with a substantial investment in very visible large mainframes and rapidly expanding data bases which were beginning to outgrow their physical facilities. Thus, the walls of the IS fortress were stormed with the demands to obtain the value from the investment.

- Thus, the productivity problem was born. If only the IS function could develop systems fast enough, the value of the investment in hardware/software systems and data bases would suddenly appear like a genie out of a lamp. This was the productivity problem which was analyzed in INPUT's major study Improving the Productivity of Systems and Software Implementation which was published in November 1980.

- In other words, productivity improvement in the systems development function was the key to all other productivity improvement efforts because the hardware/software data base systems were available to solve the productivity problems of the office. Essentially, the IS function was identified as



the bottleneck on the road to the "information age," and the role of scapegoat was masochistically accepted by the management of many IS functions. This was all before the personal computer appeared in business offices and tended to confirm the judgment which had been made about the productivity problem and also to offer some alternative solutions.

- Essentially, the problem which INPUT identified with improving productivity of systems and software implementation in the 1980 report was a management problem. The productivity hierarchy (mentioned previously) was a way of saying the real problem in software productivity was not going to be solved by any tools and aids unless a comprehensive plan for productivity improvement was in place. It was felt that this placed responsibility for improved software productivity squarely where it belonged—with IS management. Since the IS function had (probably unfairly) been identified as the primary problem, the fixing of responsibility provided both an opportunity and a challenge to IS management to become actively involved in a major business problem—office productivity in general.
- Unfortunately, the PC has tended to confuse the issue substantially and offer "alternatives" which may prove inimical to solving the software productivity problem. When INPUT addressed the productivity problem in 1984 (see Market Impact of New Software Productivity Techniques and New Opportunities for Software Productivity Improvement), it was found that the primary approaches being taken to software productivity improvement were the following:
 - Standalone PCs with user-friendly software were tending to confirm that central development facilities were truly unresponsive. Armed with spreadsheet software, end users were able to get "answers" to "what if?" questions much more readily than they ever could going through the systems group.



- Of course, the fact that they normally exercised their productivity tools by keying in data which was the output of a mainframe computer system also confirmed the inadequacy of the mainframe "solutions" which were already in place. Now that the productivity tools were available (PCs), there was the immediate cry for access to corporate data bases through micro/mainframe links.
- In response to the appearance of personal computers in the office, the central IS function was also prompted to establish information centers where qualified systems people would provide advice, counsel, and assistance in selecting and using PCs, data bases, and other productivity tools such as 4GLs.
- The new tools and approaches to systems development also facilitated prototyping of applications. Users could see results rapidly and be directly involved in the systems development process.
- INPUT concluded that all of these developments (standalone PCs, micro/mainframe links, information centers, and prototyping) had one thing in common--some of the responsibility for software productivity was shifted from the IS function to end users. INPUT kindly referred to the whole trend as "distributed systems development" (DSD), and a number of potential conflicts in the DSD environment were identified:
 - Top-down systems design does not necessarily interface well with "bottom-up" development.
 - Security of data bases is not necessarily compatible with ready access to corporate data bases.
 - The very user friendliness of PCs could be compromised by the necessary functional capability which would have to be added to permit applications of any substance to be developed.

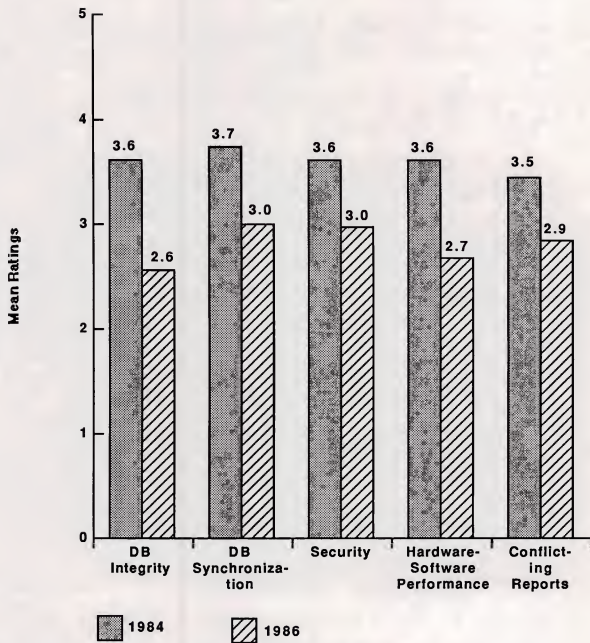


- The quality of data in a highly centralized environment cannot necessarily be assured once it is distributed for casual use.
 - Off-loading of data to support the DSD environment might increase demand for mainframe processing power substantially more than any offsetting distribution of processing from the mainframe. (Total cost of processing would rise despite the better price/performance of PCs for some processing tasks.)
 - Conflicting management information would result from the DSD environment.
 - Complex parallel trends of centralization, integration, differentiation, and mechanization which are inherent in the DSD environment (essentially networking) would make hardware and software planning even more difficult than it already was.
- Essentially, these concerns net down to the following potential problems: data base integrity and synchronization, privacy and security, hardware and software performance, and conflicting management information (reports). Respondents were asked to rate these potential problems in both 1984 and in this year's study (see Exhibit IV-8). There was a uniform and significant drop in the concern that the respondents expressed for these potential problems between 1984 and 1986. There are several possible explanations for what currently appears to back relatively modest concern for what INPUT considers to be rather serious problems in the DSD environment:
 - The current study concentrated on managers in the development center, whereas the former was directed toward more senior management (directors of MIS). It is assumed that the more senior managers would be more concerned about the potential problems inherent in the DSD environment.



EXHIBIT IV-8

RATINGS OF DSD PROBLEMS
(1984 AND 1986)



Rating: 1 = Low Importance, 5 = High Importance

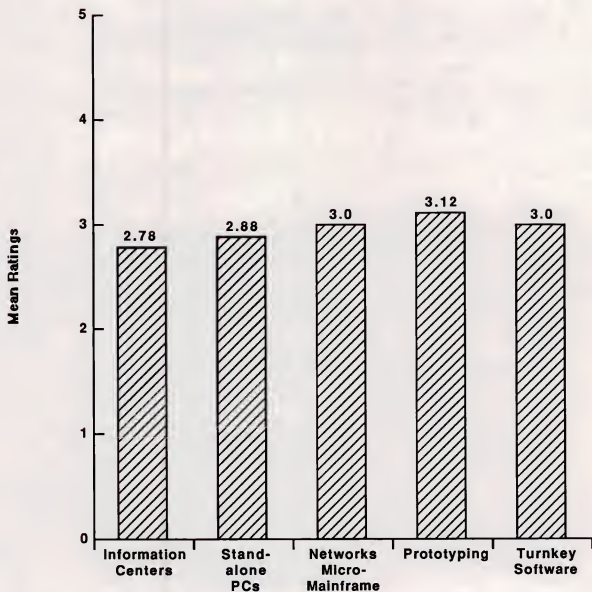


- Micro/mainframe links were all the rage when the 1984 study was conducted. Since then, not a great deal has happened in that area (now departmental processors have become all the rage). It is possible that distribution of any significant data bases to PCs is no longer considered to be nearly as much of a problem as it was originally thought.
- It is possible that the DSD environment is being created with such enthusiasm that the problems seem negligible by comparison.
- The respondents were asked how effective they felt various approaches to DSD were on a scale of 1 to 5 where 1 was defined as not being effective at all and 5 was defined as being very effective. The results do not indicate any great enthusiasm for the effectiveness of the DSD environment (see Exhibit IV-9).
 - The responses seem to cluster around the very low end of the "normal range," with information centers and standalone PCs being rated somewhat negatively with mean ratings which are less than 3.0. This is probably explained because the interviews were conducted in the development center, where it is unlikely to find very much enthusiasm for either the information center or PCs.
 - Since prototyping is billed as being an aid for the systems development function, the highest rating of 3.12 is not surprising, but it is lukewarm support at best.
- The respondents were also asked which of the following statements most nearly expressed their feelings about the general effectiveness of distributed systems development with the following results:
 - "The applications developed outside the development center are more costly to the company over the systems life cycle." Five respondents indicated that this statement most clearly expressed their feelings.



EXHIBIT IV-9

EFFECTIVE APPROACHES TO DSD



Rating: 1 = Not Effective, 5 = Most Effective.



- "The investment in end-user computing could be put to better use in the central facility." Five respondents indicated this statement best expressed their feelings about DSD.
 - "The investment in end-user computing is worthwhile because it frees the development group to work on more important work." This was by far the most frequently selected statement, with 15 respondents choosing this as being most representative of their feelings.
 - "There is a marked improvement in productivity for developing all types of applications." Six respondents were enthusiastic enough about the results of DSD to approve of this statement.
 - "Our use of distributed systems development has reduced the number of professional analysts and programmers we need." Only one respondent would agree that this was the case.
 - Assigning rating of 1 to 5 for the five statements, the mean comes out to 2.78 which confirms the somewhat negative reaction which was detected in the rating question itself. The consensus seems to be that as long as DSD keeps the users out of the development team's way it will be tolerated. However, the endorsement given to the first three responses clearly indicates that the fortress mentality which evolved during the 1970s is far from dead.
- Analysis of an open-ended question which asked whether the respondents had any additional comments concerning the potential problems of the DSD environment and what they were doing about them revealed the following:
 - Those giving low ratings to the potential problems of the DSD environment were those companies which had not done very much to establish such an environment, i.e, set up information centers, install PCs, etc.



- Among those companies which had become actively involved in implementing the DSD environment, the problems (such as data base synchronization) were very real, and most respondents admitted they did not have ready solutions.

- One measure which has frequently been quoted as a measure to indicate the magnitude of the productivity problem has been the backlog of user requests for systems. While backlog is far from an accurate measure and is subject to rather unusual interpretations (e.g., there is some feeling that as productivity increases the backlog will increase also because users will ask for more), the results of our study indicate quite clearly that whatever it is, the backlog is increasing (see Exhibit IV-10).

- Users were asked whether the backlog had increased or decreased since 1981, which was the year following INPUT's major productivity study. The results were not encouraging:
 - Twenty of thirty-three respondents said the backlog had increased.
 - Eight stated it had remained the same.
 - Only five stated that the backlog had decreased in the last five years.

- The 20 respondents who stated the backlog had increased stated that the increase over the last five years had been an average of 73%. The five who said the backlog had decreased reported an average decrease of 44%.

- A quick analysis of the five respondents who reported that their backlogs had decreased provided the surprising information that three of the five did not



EXHIBIT IV-10

**BACKLOG ANALYSIS
(Since 1981)**





have any productivity tools installed (DBMSs, design methodology, 4GL, or applications generator). This promoted a somewhat closer look which revealed the following:

- Only six of the respondents to the survey indicated that they did not have any productivity tools installed. Three of those (as stated above) reported decreased backlogs, one reported the backlog remained the same, and two reported an increase in the backlog.
- Of the two respondents reporting a decrease in backlog and the use of productivity tools, one used two tools (IDMS and ADS/Online) and reported a 50% decrease in backlog. The other reported the use of three tools (ADRS, Nomad, and Mantis) and only a 10% decrease in backlog.
- Of the eleven respondents having IBM DBMS products installed:
 - . Nine reported that their backlogs had increased.
 - . One reported it had remained the same.
 - . One failed to answer the backlog question.
- Of the other 12 respondents having DBMSs installed:
 - . Six reported backlogs had increased.
 - . Four reported they had remained the same.
 - . Two reported their backlogs had decreased.
- Of the 19 respondents who reported they had 4GLs installed:
 - . Ten reported that their backlogs had increased.

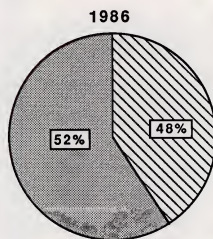
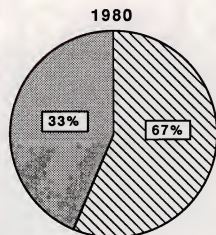


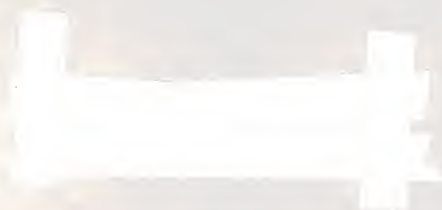
- Five reported that backlogs had remained the same.
 - Two stated they had decreased.
 - Two failed to answer the backlog question.
- What does all this mean? First of all, it is probable that all of those backlog figures which have been quoted over the years are of questionable value in measuring anything. Secondly, the productivity "solutions" to the backlog do not seem to be working very well. Finally, there does seem to be some support for the opinion that the DSD environment does, in fact, increase demands on the central development function. When asked to what they attributed the increased backlog, most respondents stated that users were becoming more knowledgeable and therefore more demanding. Essentially, this means that users in the DSD environment cannot understand why what they consider to be simple requests take so long, and they will not take "no" for an answer.
- While the backlog has been considered to be an indication of the need to improve productivity, the heavy maintenance load has been considered to be a primary cause of the unresponsiveness of the central IS department. There are indications that the DSD environment does, in fact, reduce the perceived cost of maintenance from the perspective of the development center (see Exhibit IV-11). This is understandable for the following reasons:
 - Routine maintenance in terms of report formatting and minor changes can be made by either the information center and/or end users.
 - Prototyping leads to the classification of more work as development. (The definitions of development and maintenance have never been too clear in most companies anyhow, with the term enhancement being a classification which can go either way.)



EXHIBIT IV-11

**MAINTENANCE
(1980 and 1986)**





Some productivity tools, such as 4GLs, encourage (or even require) complete reprogramming rather than modifications (maintenance).

- It should not be assumed that the perceived reduced cost of maintenance means that any major structural change has occurred in the relative life cycle costs in most companies. In fact, there can even be serious questions raised concerning the desirable ratio between development and maintenance. For example:
 - If a high-quality system is installed and it is easily maintainable, its life cycle may be longer than a system which may require replacement relatively early. The maintenance costs of the first may be substantially higher than the second, but it would certainly be the most cost-effective.
 - Heavy development costs may indicate that it was difficult to get a system tested and into production, whereas another system went into production rather smoothly and started accruing maintenance costs early in its life cycle.
 - There can be heavy development activity while merely redoing existing systems and making them more expensive to operate. Sometimes the decision is made to develop a new system because no one wants to maintain the old one. The desire of systems people to always be working on something new is not necessarily the most productive use of company resources.
- INPUT's concern about the DSD environment has been the distortion of what we consider to be the priorities for establishing a plan for a truly productive environment. Specifically, these concerns are as follows:
 - The DSD environment seems to be the antithesis of commitment to quality. All of the emphasis seems to be on immediate results with little concern for the development of a quality system.



- While end-user involvement would seem to be a plus for the DSD environment, the results of this study seem to indicate that the development center views the primary benefit to be the fact that end users are kept so busy doing their thing that they do not have time to "interfere" with the major development projects. That is precisely where end users should be involved.
 - Broadbased management of projects is not necessarily encouraged by the DSD environment. In fact, there is the potential for establishment of even more contention for resources. When end-user departments are attempting to do their own thing with departmental processors and personal computers, it will become increasingly difficult to get commitment for the support of major projects.
 - The tools and aids of the DSD environment are directed toward personnel who do not have a systems orientation. This results not only in the training problems which are already apparent, but also means that more inexperienced personnel are beginning to become active in systems development. These personnel are not necessarily effective in developing high-quality or even acceptable systems.
 - Finally, the emphasis remains on magic solutions to the problem of developing systems. There is absolutely no evidence that any current tools and aids are an effective substitute for thorough problem analysis, good design, or high-quality personnel.
- In addition to all of the above, there are some serious questions as to the impact of the DSD environment on productivity in the broader sense of the word. It is INPUT's belief that these questions are beginning to concern executive management, and that is probably the reason for the infamous computer industry "slump" which has occurred. And, it is our opinion that the situation has the potential to get worse if the proper priorities are not restored in the productivity hierarchy (see Exhibit IV-12).



EXHIBIT IV-12

PERFORMANCE IMPROVEMENT
AND DSD

PERFORMANCE LEVEL	IMPACT OF DSD ENVIRONMENT*	
	1960s	PROJECTED
Hardware - Software	-2	-2
Human/Machine Dyad	?	0
Work Units	?	-1
Institutional	?	-1

* -2 = Strong Negative Impact

-1 = Some Negative Impact

0 = Neutral

+1 = Some Positive Impact

+2 = Strong Positive Impact



- At the present time, there is no question that the impact of the DSD environment is resulting in increased demands for computer hardware/software which are out of proportion to the measurable benefits which are being received. There is every indication that we have reached an Alice-in-Wonderland stage where it becomes necessary to run harder to stay where we are.
 - At the human/machine dyad, there are serious questions as to whether professional time spent sitting at a keyboard is really cost-effective or more productive for non-systems personnel.
 - At the work unit level, it is questionable whether the increased paper volume in offices (regardless of how pretty the reports are) is a sign of improved productivity or merely a symptom of information overload and an actual lowering of the quality of the contents.
 - At the institutional level, questions are being raised about the actual impact of computer/communications technology. If the banking study mentioned in the previous section is reasonably correct, these are legitimate questions, and responsible executives are beginning to ask such questions.
- All of the above are questions which individual organizations must answer for themselves, but it should be apparent that computer/communications networks are no longer accepted as an automatic solution to office productivity. INPUT intends to explore these issues in the Distributed and Office Systems Directions report series which will be introduced in 1987. However, without knowing what the actual situation is today, it is certainly not difficult to predict that there are inherent dangers in the DSD environment which is being established in an attempt to improve software productivity (see Exhibit IV-2).



- We cannot keep throwing hardware and software at the productivity problem without eventually impacting performance at other levels. The projected impact of the potential problems associated with the DSD environment which are presented in the exhibit are actually conservative, but there is no question that it is possible for the impacts to turn negative.

- The impact at the human/machine dyad is generously assumed to be neutral since it is assumed that responsible humans will only use systems that do not seriously impact their productivity. Sooner or later, end users are going to insist that they have ready access to data rather than having to key it in and will get tired of manually handling data bases, and if the technology does not support their particular function they will not use it. However, the volume of information available on the network will increase to the point where they will be required to spend an increasing amount of their time handling both paper and electronic information which does not necessarily enhance their personal productivity.

- INPUT's emphasis has been on the fact that more information is not necessarily better and that the quality of information has a definite tendency to deteriorate in the DSD environment unless attention is returned to quality. The research for this study indicates that there is little concern on the part of systems developers for the quality of information flow, and it is our belief that unless professional systems people become concerned, the productivity of work units will be impacted by the sheer volume of information which is available. (An undue amount of time will be spent reconciling and arguing about conflicting information.)

- Degenerating information quality and increased computer/communications costs can have a decidedly negative impact on institutional performance, and the potential for disastrous and even catastrophic



impact are there in certain operating environments. (Some companies may not be able to compete because of investments in systems which fail or do not deliver promised results.)

- The case studies which were conducted for this report tend to substantiate these conclusions concerning the DSD environment.

C. CASE STUDIES

- Three of the case study organizations are those which have been used in previous INPUT reports on operating systems and departmental software. The general characteristic of these organizations is as follows:
 - Case Study #1 is an organization which closely follows IBM's perceived operating system and networking strategy.
 - Case Study #2 is an organization which has made major modifications within the IBM operating system architecture and is attempting to maintain a presence in "both camps" between the IBM mainframe orientation and an equally strong (and uncontrolled) minicomputer environment.
 - Case Study #3 is an organization which integrated "administrative" and "engineering" processing into an international network.
- It is recommended that the earlier case studies be reviewed in order to understand the operating environments in more detail and to enhance the value of the current case studies.
- The other two case studies were selected specifically for this report and have the following characteristics:



- Case Study #4 is an organization which is currently planning the development of a major system which is vital to its business. Currently, there is a massive, tape-oriented system with 55 million records, and the effort is directed toward an on-line transaction-oriented system.
- Case Study #5 is with an organization which has accepted the importance of maintenance and the impact that the DSD environment has on the maintenance function.

1. CASE STUDY #1

- The organization is a major financial institution which has traditionally followed IBM's general hardware and systems software direction.
- Not surprisingly, IMS is the primary host subsystem (although it was admitted that there was "some" CICS), but DB2 is required to provide the desired flexibility for decision support. This dual DBMS strategy gets rather clumsy across systems where it becomes necessary to extract data from an IMS host "production" system and build relational tables for DB2 on a development or information center system. The mode of operation can be a batch run with magnetic tape output for reloading on the other system. The impact on "ease of use" as an operating systems objective and the implications for data base synchronization and integrity are obvious.
- But the story does not end there. At the end-user level, for reasons of productivity improvement, it is necessary to have a variety of other tools employed. These range from a Teradata data base machine down to word processing packages on personal computers.



- For this study, the vice president of end-user computing was interviewed. He is responsible for internal and external timesharing services, personal computers, and office automation. The purpose of the end-user computing department is as follows:
 - To review hardware and software products for three operating environments:
 - Mainframe timesharing systems (as described above).
 - Minicomputer systems (primarily Wang or IBM word processing systems).
 - Microcomputer systems (Compaq PCs, various IBM PCs, Apple Macintosh 512s, and Wang PCs).
 - To classify all reviewed products in one of four categories:
 - Standard "A," in which case the department actively markets the product to internal users and provides full support (acquisition and installation assistance, training, and documentation).
 - Standard "B," in which case the department provides acquisition assistance only if a Standard "B" product is not available (other support is not provided).
 - Not Recommended, in which case no support is provided because a better product has been found (or it is found that the product does not satisfy user needs).
 - Under Evaluation, in which case no support is provided until the product has been evaluated.



- The obvious intent of end-user computing is both to provide assistance and to exercise control over the hardware/software selections of the end-user community. To give some indication of the magnitude of this undertaking, the first end-user computing "Product Catalog" contained over 100 "Standard A" products and a "Master Chart" of nearly 400 products which had been, or were being, evaluated.
- There are six internally developed "financial analysis and modeling products" and 10 commercial products (such as Lotus 1-2-3) listed in the catalog.
- Several "reference (proprietary) data base facilities" are supported by the end-user computing department. These require training and qualification of both internal and external sources of the data.
- There are 11 DBMS and "information retrieval systems" products listed in the directory. Four were developed by the bank (primarily enhancements to Nomad2), and seven came from external sources.
- There are 10 graphics packages supported by end-user computing, including those integrated with packages such as SAS and Lotus 1-2-3.
- The text and word processing packages supported are primarily Wang-oriented (12 of the 16), but two packages are fully supported for the IBM PC and compatibles (MULTIMATE and VOLKSWRITER).
- Despite having internally developed office support systems (personnel directory, electronic mail system, and mailing list system), the bank also installed PROFS and found that it has "a life all its own."
- Two statistical analysis packages are supported.



- Twenty-six systems products and utilities are supported from major operating systems such as CMS under VM/SP (the mainstream operating systems are MVS/XA and TPF supported by the central systems programming group) down through a myriad of file transfer (and backup) programs for moving files around among the various mainframe-, minicomputer-, and microprocessor-based systems.
- MS Basic is supported for "applications development" on micro-processor-based systems, and Nomad (heavily modified) is supported for mainframe applications development by end users.
- Cross-functional interfaces include a virtual network for transferring files across machines (and the output devices attached to those machines) and between Wang operating systems (OIS and VS) and the mainframe timesharing environment.
- Regardless of the overlap of specific products in various categories (such as Lotus 1-2-3), the cost of supporting just the "Standard A" products listed above is obviously substantial, and the additional cost of evaluating products is, in itself, awesome. For example, when the publication was published, over 50 products were under evaluation, ranging from DisplayWrite 3 to Wang Office and SYMPHONY, TIMM (The Intelligent Machine Model for building knowledge-based systems), and hardware such as the IBM LAP PORTABLE and the PC AT.
- Most of the products listed in the product catalog could be classified under the general umbrella of "departmental software" even though there are currently relatively few "departmental processors" installed. (The bank has made a concerted attempt to stamp out minicomputers in line with its "true blue" operating systems orientation.) These products are all designed to improve white collar productivity from typists to loan officers managing multibillion dollar portfolios. The bank is currently expending more resources evaluating, installing, and supporting "solutions" for end users than most



companies spend on problem analysis and implementation of the systems (both computer- and paper-based) necessary to run their organizations. The end-user computing department tends to focus attention on the end-user "revolution," and several conclusions can be reached from its experience:

- The variety of "solutions" (products) available far exceeds the defined problems of the bank.
 - Even the most comprehensive office products address only a small portion of the white collar productivity problem and would more properly be classified as being "tools" rather than "solutions."
 - The availability of information processing tools immediately identifies the major missing ingredient in the "solution"--data.
 - The ability to transfer data and files among various hardware/software products on the network (the bank has invested substantial resources in providing these facilities) soon reveals there is no assurance that the quality of information generated by the process will be sufficient to produce a measurable improvement in white collar productivity.
 - Indeed, it is felt that the quality of information could decrease substantially (despite rapidly increasing investment in hardware/software technology) if it were not for the end-user computer department. In other words, it is necessary to expend an enormous amount of effort merely evaluating and supporting the various "solutions" available in the marketplace if chaos is to be avoided.
- In fact, it becomes difficult to realize that chaos has been avoided. When we last visited the case study bank, they had installed IBM token ring LAN and were reaching the conclusion that data base administrators (or LAN supervisors) would be necessary in order to make effective use of the available systems. Work units are complaining that they do not have sufficient



information to manage their loan portfolios, and the bank is going through the most difficult time in its history. It has tried practically everything IBM has recommended (except the use of System 36s as departmental processors) and has one of the more advanced environments for distributed systems development, and nothing has worked.

- While provision has been made to distribute data from corporate data bases, these data bases themselves are in question. While there are 17 corporate data bases of record under IMS, on investigation this grew to 40 data bases of record and then to over 70. These "other" official data bases are under "practically every" access method, and reconciliation problems at any given point in time can run into the billions.
- Despite the fact that the case study company has invested enormous amounts in computer hardware and software productivity tools and has had a number of outstanding systems personnel give it their best shots, the quality of data and information available for running the business appears to have degenerated over time. Fundamentally, the problems the institution have had are systems problems which have been exacerbated by using the very tools which have been proclaimed to be solutions.

2. CASE STUDY #2

- The second case study organization is a leading university which was a sponsor of INPUT's major multiclient study on productivity and which has the following characteristics:
 - The university is on the leading edge of computer/communications technology, having served as an early testbed of LANs, and it is currently installing an integrated voice/data network on campus.
 - Computer literacy is an active program among faculty and staff, and "work at home" is encouraged over the public network.



- Proprietary IBM operating system enhancements are complemented by an advanced DBMS (the equivalent of a 4GL is integrated with the DBMS), electronic mail system, text processing system, and a graphics and printing system. (These services are actively promoted through the central information technology services department.)
- These central services are IBM host-oriented, but minicomputers (primarily DEC) are installed in many departments (frequently in connection with specific research projects or grants). As mentioned in the operating systems report, this has resulted in the central processors being viewed as a node on Ethernet, and the manager of systems programming stated, "they try to keep a foot in both camps" by providing access to both DEC and SNA terminals.
- However, the office automation effort on the campus has been placed under administrative information services which is essentially oriented toward a large-scale IBM mainframe. Due to the nature of the academic environment, very little effective control can be exercised over the selection of particular hardware/software products used by the various schools on the academic side of the university.
- Therefore, the acceptance and use of productivity tools over the university's network hierarchy provides some insight into their usefulness. Fortunately, departmental information systems had just completed a study of office automation on the campus, and while it was not designed to focus on software productivity issues, it does permit certain conclusions to be drawn.
- The word processing and DBMS developed by the university were installed long before personal computers became available, and an extensive electronic mail system has been available for a number of years. The recent survey revealed the following:



- There is a trend to move off the central mainframe systems for both word processing and DBMS despite the availability of well advertised and promoted systems on the mainframe which have richer functional capability. (The DBMS system was established as a standard for all systems development a number of years ago.)
 - The EMS system which was made available free for one year has seen use drop off since it has started to be charged back to users.
 - The graphics and printing system is one of the most advanced in existence, and it is used extensively (as would be expected considering the publishing which is done on the academic side). However, the service was provided at a substantial loss during the most recent academic year.
-
- The reasons given by the defectors from the central system are "don't need it" and "it costs too much." The study concluded that the failure to use the central tools was regrettable because the real payoff normally came from the integration of "individual, departmental, and central administrative systems (institutional)," and this was possible only through the central facility.
 - The central IS function's view of the DSD environment is that the off-loading of word processing is understandable because of cost, but the somewhat limited use of the EMS system is difficult to explain because the systems department itself finds it very useful and it has been actively promoted on campus. Spreadsheet packages are considered to be error prone, and they are attempting to work with users to standardize their applications. As far as data base systems on PCs are concerned, it was stated that as soon as end users want to build data bases, they must receive extensive systems training.
 - The administration of the university is concerned enough about the problem that a "team for improving productivity" (TIP) has been set up to review the



problem. This team started nearly three years ago to explore the "under-utilization" of hardware (at all levels in the processing hierarchy) and the tools and services which had been made available.

- The team started by making hardware inventory in order to find out what was really "out there" and was surprised to find how much processing power was scattered around.
- Then it was discovered that while there was hardware and software not being used, there were questions about the true productivity impact of computer use on managerial and professional personnel. Services which were thought to be valuable such as EMS were going unused, but some administrative personnel spent abnormal amounts of time rekeying data into PC software packages. Paper remained the primary information flow across the organization.
- The TIP group confirmed the IS group's conclusion concerning the use of PC software (word processing, spreadsheets, and DBMS), but the true impact on productivity became further clouded by the following:
 - The concern about managers and professionals sitting at key boards.
 - The feeling that the increased length and volume of communications made even word processing suspect for productivity improvement.
 - The true cost of training end users in the effective use of the systems was far higher than anyone wanted to admit (much less pay for).
 - The maintenance of personal data bases was viewed as becoming a problem quite rapidly, and they concurred with the central IS



function concerning the need for systems training for those setting up data bases.

- Then, when the TIP team got to the departmental systems (which for the reasons previously given are uncontrolled nodes on the network), they discovered that in order for them to be effective they required dedicated personnel--analysts, programmers, or something.
 - The TIP team is becoming convinced that the best available hardware and software tools do not necessarily lead to better information unless professional systems people are involved. And, the team concurs with the study in its finding that integration across the three performance levels (individual, departmental, and institutional) is necessary for real payoff and that this does not come automatically.
 - In addition, there is concern that the trend toward "iterative programming" (prototyping) in the systems area may lead to problems with both internal and external auditors. And, the team has been alerted to the fact that security is a "terrible problem" which can only get worse as the network grows.
- The information technology services department has come to the following conclusions concerning the role of a development center:
 - The problems of data structuring are much more important than programming.
 - The most important developments for improving productivity over the next five years will be:
 - Expert systems (the university is a leading center for artificial intelligence).



- Micro/mainframe "cooperative processing."
- And, "perhaps the most important of all," an understanding of the information process.
- The role of the central development function will change with less time being spent on development activities and proportionately more time on quality and general consulting.
- Generally speaking, the university is taking the same broad assessment of productivity improvement that INPUT has recommended. Since 1980, they have had a number of years of heavy development activity during which most of the major systems have been replaced, including a major hospital system which was purchased. Currently, systems staff time is split approximately 25% development and 75% maintenance, and increased effort is going into making distributed systems development work. Office productivity is recognized as being of primary importance, but the cost of delivered services is coming under close scrutiny.

3. CASE STUDY #3

- While the productivity strategies of Case Studies 1 and 2 may seem quite different, there is a striking similarity in sharing IBM's aversion for minicomputers in the processing hierarchy. The third case study involves a semiconductor company which, while maintaining an IBM mainframe orientation for administrative processing, has established an interconnected network of DEC minicomputers.
- The director of corporate information services (CIS) was first interviewed for this study shortly after his department was featured in the in-house monthly newspaper (March 1986). The first paragraph of the article was as follows:



- During the past few years, CIS has been diligently turning localized, manual systems throughout the company into global, electronic systems. These advanced, interconnected information systems are bringing us closer, step-by-step, to the highly efficient, 'paperless' society of the future."
- This clearly stated objective of eliminating paper-based manual systems with computer/communications networks addresses information flow within the company. It goes far beyond current office automation systems which seem to concentrate on producing printed documents more efficiently.
- INPUT believes that this represents a solid foundation for a comprehensive productivity plan.
 - Fundamentally, a commitment to quality has been made and the general direction has been established. In a not too subtle fashion, it states that the company is going to change the way it has been doing business. Computer/communications networks are going to replace what has been the primary information flow within the company-- paper.
 - Since end users are normally responsible for paper systems and procedures, it is obvious that they must become involved in replacing those systems. End-user involvement is assured as soon as the director of CIS commits to replacing the operational systems which are currently being used to run the business. In the published article, this commitment and challenge was stated as follows:
 - "When you consider that over the past two years we've grown from about 400 terminals to 1,400 terminals and that dozens of slow, manual systems that had been plugging along for years have been replaced with highly efficient electronic systems that

1. The first part of the paper discusses the importance of the study of the history of the English language.

2. The second part of the paper discusses the importance of the study of the history of the English language.

3. The third part of the paper discusses the importance of the study of the history of the English language.

4. The fourth part of the paper discusses the importance of the study of the history of the English language.

5. The fifth part of the paper discusses the importance of the study of the history of the English language.

6. The sixth part of the paper discusses the importance of the study of the history of the English language.

7. The seventh part of the paper discusses the importance of the study of the history of the English language.

8. The eighth part of the paper discusses the importance of the study of the history of the English language.

9. The ninth part of the paper discusses the importance of the study of the history of the English language.

10. The tenth part of the paper discusses the importance of the study of the history of the English language.

11. The eleventh part of the paper discusses the importance of the study of the history of the English language.

12. The twelfth part of the paper discusses the importance of the study of the history of the English language.

give people all over the world input and access, you can see the way our world is changing."

- "Information is becoming more accessible, and there is a need to take strategic advantage of that information."
- Broadbased management is also virtually assured by clearly and publicly stated goals. If management does not support the goals and objectives, they have an opportunity to become involved. The fact that the number of terminals supported on the network more than tripled during a two-year period when the semiconductor industry was in a major slump clearly indicates the support of management at all levels in the company. As the director of CIS said: "High tech has got to go high tech. If we don't, we're dead."
- In an environment which has the base of the productivity pyramid in place it is not difficult to attract and retain effective personnel. In fact, it practically goes without saying that an organization which has a comprehensive productivity improvement plan and is making progress toward its goals has effective personnel.
- Given a comprehensive productivity improvement plan, the right tools/aids/methodologies practically become an afterthought.
- The development environment which has been established can be generally defined as follows:
 - IMS is the primary DBMS, and it is not anticipated that it will ever be replaced. When asked about DB2, and its key role in IBM's distributed data base strategy, the response was: ". . .if we had waited for IBM to build our network, we would still be waiting. In fact, if we had waited for anybody to build our network we would still be waiting, and we can't wait."



- TSO has been used to develop the 17 major commercial systems, and the primary objection has been the heavy overhead associated with its use. FOCUS (under VM) has been used to develop 250 "end-user systems," and they are reasonably well satisfied with it. However, the director of CIS wants to see all development on micros. ("You don't need a 3090 or even a VAX to develop systems.")
- Multiple languages are employed within the company, and Cobol, PL/I, Fortran, and Pascal are all supported by CIS.
- They are committed to DIOSS on their LANs and, when first interviewed, the director of CIS was waiting for IBM to announce a "4300 micro" which would be VM/DIOSS-oriented and could be used as a departmental processor. Since IBM announced the 9370, he still does not feel that IBM has as much understanding of distributed processing (departmental processors) as DEC does.
- Electronic data interchange (EDI) with vendors and customers is of major priority, and they are now in the process of determining the hardware and protocols which must be supported in order to satisfy most of these requirements. (The importance of communication standards is recognized, and the director of CIS decided years ago that IBM could not be expected to provide necessary solutions to his problems. He spends considerable time on industry efforts to address communications problems and establish standards.)
- Very little thought has been given to PC systems software. (There seems to be the general attitude that the top-down approach under the greater SNA umbrella will eventually encompass them anyway, and who cares what they are doing when they are not connected.)



- Among the other tools employed are a concentration of products which address performance, accounting, and security weaknesses with IBM systems. Performance and security on the network are recognized problems and are scheduled for improvement under the CIS plan.
- The focus within the company is clearly on network management and not the operational problems within individual processors (or processor complexes) except in so far as they represent bottlenecks to information flow. Generally speaking, little attention is given to minicomputer operating systems and no improvement project is scheduled on the DEC systems. However, there is a specific plan in place to "upgrade/improve MVS-SNA software." This plan states:
 - "This project involves upgrading installed software products to current levels and releases plus adding two products to provide data for network utilization and performance tuning."
 - "The MVS-SNA network has reached the practical limits of the software installed. Replacing/upgrading the software will provide the safety required to avoid having the software environment restrict business expansion and growth and will keep (us) current."
 - The message here is very clear, the company does not want software tools (in this case, IBM's networking and operating systems strategies) to be the limiting factor on the expansion, growth, and flexibility of its business.
- The fourth case study seems to support the statement made by Ken Olsen (CEO of DEC) at the time VAXmate was announced: "The problem is that everyone's been going about this backwards--buying lots of computers and then trying to connect them together. We have to start thinking about the computers as peripherals. You start with the network, then you hang the computers on later." The director of CIS would certainly seem to agree with



this statement, and it would appear that a great deal of progress can be made by not going about it backwards and addressing the problems of the nodes before building the network. Perhaps the greatest productivity tool of all is to start addressing information flow before building enormous central data bases and to provide everyone with tools to generate any information they think they want.

4. CASE STUDY #4

- The fourth case study is of a major publishing firm which is in the process of converting a major batch system which is primarily tape-oriented. The file contains 55 million records and is one of the most complex mailing list applications in the world. At the present time, the more complex logic of the system is written in assembly language. The system is the key to the publishing firm's business plan. Its characteristics are as follows:
 - The system presents a "mosaic of people and residences coming together and separating over time." The primary objectives are to maintain an up-to-date list of specific people and their residences, so that an annual promotional effort can be accurately addressed. Based on long experience, the company knows the following:
 - The number of subscriptions which will expire each year.
 - The percentage of new subscriptions which will be received from the population of people who receive correctly addressed promotional material.
 - The mailing list is actually the key element in the company's business plan. Adding new prospects to the mailing list (and tracking those who move) is the means by which the company maintains its sales and grows. The list is maintained from various sources including state motor vehicle registration lists and outside mailing list suppliers,



although these are used primarily for cross checking their own list, which is usually more accurate. Use of the list peaks during a promotional period which spans several months. The system is an operational horror. Once the mailings start, you have reached the point of no return, and the quality of the list determines the success of the campaign (barring any operational failures).

- The plan is to move the system from its current serial batch orientation into the wonderful world of DBMS, and IMS has been selected. The system will be programmed in COBOL. Originally, the plan was to have a single project extending over several years and then cut the system over. However, a phased approach is now planned with the first phase to be operational in January 1988. Even the phased approach becomes extremely complex because of the size of the data bases which must be maintained and the many sources (and formats) of outside data which are used. There will be a requirement to maintain the view of data as a sequential tape file for an extended period of time.
- Despite the complexity of some data analysis and logic built into the programs, the real problem is one of data structuring and the volume of processing which will be required to build, maintain, and operate the data base. The project team considered a number of applications development tools and selected Knowledgeware because of its rigor in data definition. The experience to date has not been entirely satisfactory because of some "little technical bugs," and the fact that the ability to consolidate different parents and roots does not yet exist on the mainframe. (Evidently, the ability currently works from PC-to-PC but is only going into Alpha test on the mainframe.) Nevertheless, it is felt that such a facility is essential for the development effort, and the company will continue to test the facility.
- Having already established an information center, the company has used IMS for screen design and RAMIS for report generation in the past. However, these are not considered appropriate for major applications development. The



merger of Knowledgeware with Tarkenton caused some brief reconsideration of Gamma as an applications development system, but this was not taken too seriously because applications generators are not considered to be suitable for the complex logic which is required in the mainline applications.

- In fact, as the conversion project goes on, there is increasing concern about potential hardware/software performance problems which may be associated with using IMS for very large operational data bases. The case study company has asked to talk with other organizations which have used IMS for comparable applications and, as this report was being completed, they have not received any satisfactory references. During the course of their inquiries concerning IMS performance with very large data bases, the subject company has uncovered some additional causes for concern:
 - They learned that on very large data base development projects, fully half of the development effort involved preparation and running of test cases against the data base(s). This was completely out of line with their past experience with development projects, as well as with the 19% of development effort which has been fairly consistently reported for testing and debugging on "normal" development projects.
 - When IBM was unable to supply reference locations for very large IMS data base projects, the case study company made discrete inquiries concerning IBM's internal use of IMS. It was discovered that IBM does not use IMS for some of its largest and most critical applications. A case in point being the IBM order entry system which was developed using many BDAM files, and which operates at transaction rates which are six to seven times those which can be achieved by IMS for comparable processing.
 - It was also learned that IMS may not have the flexibility they are seeking for the system they are planning. However, considering there are potential performance problems even with IMS (supposedly the



DBMS for operational data bases), it is felt that DB2 is not a practical alternative. Therefore, the project will proceed with all due caution.

- There is no question that data base design and implementation are the primary problems which confront the developers of major new commercial applications systems. The tools for design, implementation, and management of data bases have been in existence for some time. These tools have been of great value in many applications, but it is important to recognize that there is a cost associated with general purpose systems which becomes unacceptable on central data bases of certain sizes and with certain transaction rates. Vendors are naturally reluctant to establish clear guidelines for the use of their systems, especially if they feel performance problems can be overpowered with additional hardware. Potential users who are in a performance grey area have every right to be concerned; there is nothing less productive than developing a system which just does not work.

5. CASE STUDY #5

- The fifth case study organization is a pharmaceutical company which has actively pursued software productivity improvement by reviewing and testing various products and approaches, as well as engaging outside consulting assistance in establishing a program of productivity improvement. Nearly five years ago they became quite interested in "information engineering" as it was originally defined by James Martin and Clive Finklestein. Several work units received outside training in defining information requirements based on business needs and functions, and tools provided by the consulting firm were installed. The approach was a step back from the "big bang" theory of information resource planning which had been made popular by IBM's Business Systems Planning (BSP) approach. However, it turned out that even the modified approach suffered from the same problem which a respondent in an earlier study identified by saying: "BSP is a good idea which is impossible to implement." The consulting firm which was promoting "information engineering" is no longer involved in the productivity improvement program of the case study company.



- Since then the company has pursued a rather more practical approach of attempting to get experienced systems personnel into close and continuing working relationships with specific organizational entities within the company. In addition, the same systems personnel are made responsible for development, enhancement, and maintenance of systems within their organizational areas of responsibility (as well as helping end users with software productivity tools). This approach is designed to give emphasis to the base of the productivity pyramid (commitment to quality, end-user involvement, and broadbased management) with special emphasis on being sure effective personnel are available through the recognition that enhancement and maintenance are of crucial importance and require experienced personnel.

- The interview was conducted with a systems analyst/programmer who has over 20 years of experience and has been with the case study company for seven years, the last five of which have been spent with the human resources department. She is currently active in the Software Maintenance Association because she firmly believes that maintenance takes precedence over development in assuring end-user satisfaction, which translates into a high-quality system to meet the company's information needs.

- The company currently has SDM installed, but the respondent stated very simply that: "It doesn't help; it creates excess documentation. You can get the system up and running before you can get the documentation done." Productivity tools most frequently employed are:
 - Focus, DB3, TSO, IDMS, Inquire, COBOL, and VSAM are all used for various systems and applications. This means that anyone maintaining or enhancing the various applications (or working with end users) must be "multilingual."

 - End users are primarily concerned with report generation and routine maintenance of reporting systems. Enhancement is the responsibility



of systems personnel and must be budgeted for just as development would be.

- The question was asked: "Does maintenance ever result in significant enhancement of systems, and do new applications get 'maintained and enhanced' into existence?" The answer was: "There is some tendency for this to happen, but that has always been true hasn't it? And, there is really nothing wrong with it as long as it is in the best interest of the company. I can make a pretty good argument that the overhead of development projects is the real cause of the productivity problem in the first place. If I am working closely with my users and we can keep them satisfied within the maintenance budget, that benefits both of us and the company. The fact is that systems evolve over time under any circumstances. What is prototyping except the recognition of what has been happening? If you're smart you just don't start over from scratch."
- When asked about the promised benefits of 4GLs and their impact on maintenance. The answer was just as pragmatic: "Look, there is no magic in 4GLs; they have been around for a long time under other names. They are just another James Martin thing. Have you read what Nicholas Zvegintzov (the editor of Software Maintenance News) has said about Martin? Some people take issue with him because they like Martin, but the fact remains that 4GLs have created a whole new set of problems; getting results fast isn't always the answer. Some people start over with a 4GL rather than maintain or enhance an existing application or use existing code. That isn't economical, and then you wind up with a lousy application that frequently has to be redone or replaced. We try to control end-user use of 4GLs. And, no I don't believe that applications programmers are going to disappear either."
- When asked about productivity and how it can be improved, she made the following statement: "I've been around for a while and I have programmed in a number of languages starting in Fortran, then assembler (which I preferred



for a long time), never did like Cobol, and I have used 4GLs and various data base systems; you use what is most appropriate or what someone else stuck you with. However, there is only one thing which makes me more productive--I have to know my customer's business. That is all there is to it; you can't expect end users to be systems people--that is my job. And, neither of us can be productive if I don't know their business because I am either going to take too much of their time or I will develop a rotten system. So I have to learn their business and stick around long enough to help them run it. Most programmers aren't doing that, they prefer to develop and run, frequently before the project is finished (much less sticking around to make it really satisfy the user's needs)."

- We concluded the interview with the general agreement that Gopal Kapur, a productivity consultant, is probably correct when he states that he can guarantee any company a 20% increase in software productivity practically overnight. The answer is to fire 20% of your analysts and programmers; the trick is knowing which ones.
- That is an appropriate ending for the evaluation of current solutions to the productivity problem. Practically anything will work in theory, but nothing seems to be working in the real world.









V FUTURE DIRECTIONS

A. INTEGRATED APPLICATIONS DEVELOPMENT SYSTEMS

- Both the telephone interviews and the case studies clearly indicate that the development functions in most companies still have a "productivity problem." The case studies in particular point out some of the problems which continue to confront the organizations responsible for the development and maintenance of major applications systems.
 - Case Study #1 has invested enormous resources in following conventional wisdom concerning DBMS, 4GLs, information centers, and end-user computing. In fact, the company continues to spend enormous amounts merely evaluating tools, aids, and approaches to productivity improvement. And yet, the quality of information within the company has deteriorated to the point where it has become increasingly difficult to run the business.
 - Case Study #2 has invested a substantial amount in developing its own tools and providing end users with communications and information systems which should improve office productivity substantially. However, many potential users reject the tools and services and prefer to use personal and departmental systems. Productivity and systems quality remain of continuing concern, and the effectiveness of installed systems is coming under increasing scrutiny.



- Case Study #3, while taking a network approach to information flow and having the meaningful goal of automating manual systems and reducing paper, has established parallel (albeit interconnected) networks for "engineering" and "administrative" information. Vertical integration of mainframes, minicomputers, and microprocessors has not been addressed, and the company's assumption that PCs will be integrated under the greater SNA umbrella may find them obsoleted before such integration ever takes place. (In other words, the PC systems which are currently being developed will have to be replaced along with their supporting manual systems.)

- Case Study #4 represents a slightly exaggerated case of a classic problem--at what point do generic solutions fail to solve problems (in this case, complex logic and very large data bases) because of implementation and/or operational costs. Anticipation and prediction of performance impact of various tools and aids has become a problem in its own right.

- Case Study #5 clearly points out that the use of various tools and aids actually complicates the already recognized maintenance problem. In addition, the solution to the maintenance problem--having a highly qualified, long-term programmer/analyst assigned to the end user--raises the old familiar question: "What if something happens to the person who has been responsible for these systems for the last X number of years?"

- If there is one very strong message which is coming back from the development centers, it is the fact that no single solution has been developed for the problems which have been identified with either computer/communications networks or distributed systems development. Multiple languages, data access methods, communications access methods, and operating systems become necessary when dealing with today's development environment. At the very



least, integrated applications development systems (IADS) with the following characteristics are required:

- Very high level languages should be available for process description from design specifications, and these languages should facilitate the generation of applications in various target languages (COBOL, PL/I, etc.). This implies the internal use of a meta-language or pseudo code to facilitate the support of various target languages, and the compilation of those target languages into optimized machine code.
- Terminals (intelligent workstations, PCs) should be supported with screen design and reporting facilities which are both screen driven and/or data driven. Various terminal access methods should be supported, and there should be flexibility of the target terminal system during prototyping of the application.
- Object-oriented data definition, description, and modeling capability should be available during development (and/or prototyping), and there should be flexibility during the development process as to the target DBMSs or access method(s) which will be employed in the installed system. This implies an internal data description meta-language or data schema (probably set theoretic or relational) which should also facilitate the transformation of data structures.
- While the generated application system should be independent of the development system, and therefore maintainable in the native (target) environment, the development system should also be capable of addressing all phases of both the development and life cycles. This means that testing, debugging, and installation documentation aids must be available through the IADS, making it capable of being the only development and maintenance system which is used.



- Last, but not least, the IADS must be able to generate applications which will operate in a variety of operating systems environments. This implies the generation of JCL and command structures.
- Integrated applications development systems are becoming available. INPUT recently had an opportunity to evaluate one such system (which is not currently available in the United States) which seemed to be quite promising. However, during the process of attempting to find a potential partner for the marketing of such a system, it was determined that several organizations which market specific tools (DBMS, 4GL, or utility packages) either did not understand the significance of an IADS or were only interested in a system which incorporated their particular product to the exclusion of others. This rather short-sighted attitude may mean that some companies prefer to ignore today's market requirements in order to promote yesterday's solutions.
- However, IADS are beginning to emerge, and INPUT has found that users who have evaluated and/or installed such systems normally look at the same five systems:
 - APS (Advanced Programming System) from Sage Systems, Inc.
 - Gamma (which theoretically will now be integrated with Knowledge-ware) from Tarkenton Software, Inc.
 - Pacbase from CGI Systems, Inc.
 - Telon from Pansophic Systems, Inc.
 - Transform from Transform Logic, Inc.
- While these systems vary substantially in terms of their functions, price (\$100,000 to \$250,000), age (Pacbase has been around for over ten years and Transform was only introduced in 1986), and target environments, they have



one thing in common, they were all originally developed by relatively small organizations. Again emphasizing the fact (which most of us have known for sometime) that large software organizations have great difficulty in developing high-quality, innovative software products. (As one of the software pioneers interviewed for this study has stated many times: "Let me know when you find a software system which everyone is really proud of and which was developed by a large group of people.") Past solutions become part of the problem when attempting to adjust to changing market requirements.

- While it is beyond the scope of this study to analyze IADS in detail, and they did not show up in our interview schedule (except for the case study company which had evaluated Gamma), we did do a spot survey of companies which had installed them and found the following:
 - There is normally a lengthy evaluation process, and when making such a serious commitment (both financial and technological), companies normally review a number of products. When an organization is looking for an IADS, it is normally aware of the ones that are available (those interviewed were aware of the above products).
 - Even those who are currently using an "older" system such as APS or Pacbase state that their decision to install an IADS has "paid off."
 - There is tremendous enthusiasm among some of the users.
 - One user stated that all future development work would be done in Telon.
 - A Transform/IMS user stated that applications development was done in one-tenth the time, and certain projects would not even be considered if the system was not available.



- While it is considered wise to temper such enthusiasm with the knowledge that the users interviewed were recommended by the vendors, and IADS are far from a total solution to the productivity problem, it is important to recognize that such integrated systems have value beyond "replacing COBOL" as a development language. After all, there has never been any question languages superior to COBOL are possible and even necessary in certain environments. The important fact of the matter is that IADS are necessary because we are evolving into a multi-operating system, multi-DBMS, multilanguage environment in which computers and communications and data processing and office automation must be integrated. Applications must be flexible in order to evolve with this environment; otherwise, past maintenance problems are going to appear trivial compared to what we will be confronted with in the future.

B. NETWORK EVOLUTION

- For over ten years, in hundreds of reports and presentations, INPUT has presented a "proper" hierarchical network consisting of very large mainframes, minicomputers, and intelligent workstations. Both the economics and the appropriate assignment of function have been explored in some detail. Finally, in 1986, it has been "discovered" that minicomputers do have a place in the networks. Unfortunately, probably for all concerned, the economics are being emphasized without regard for the proper distribution of function. The arbitrary distribution of applications on a decentralized basis (to either minicomputers or microprocessors) is going to present problems of data base quality control which could be disastrous for many organizations.
- With all due respect for minicomputer vendors, and we have certainly supported them on a technological basis over the years, it is not clear that they either understand or are concerned with the problems which are associated with distributed data bases. The current trend seems to be to chop as much processing as possible off the mainframe on an application-by-



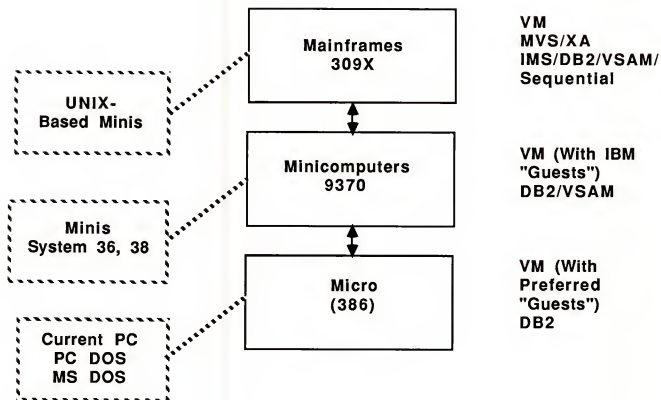
application basis. In most cases this will eventually have a substantial adverse impact on the quality of information in the organization. The trick is to distribute the applications themselves over the network (mainframe, minicomputer, and intelligent workstation) properly. Applications development tools (such as the integrated applications development system defined above) which support this proper distribution are required, and it is not apparent that many hardware/software vendors are sensitive to the problem much less intent on solving it.

- While it certainly has not been sensitivity to the problems of distributed data bases which has motivated IBM's reluctant approach to distributed processing, the highly centralized SNA approach which has evolved over the decade makes more sense now than it has in the past. With the announcement of the 9370, IBM has clearly signaled its preferred solution to the problem and that solution will become increasingly clear during the coming year (see Exhibit V-1).
- Large mainframes become big data base machines with multiple DBMSs and access methods installed. VM provides the means of integrating lower levels of the processing hierarchy and "other" operating systems. MVS/XA remains the primary "production" operating system in an environment which will, like it or not, be heavily batch oriented.
- The 9370 is a minicomputer by INPUT's definition (processor costing less than \$200,000). We have never taken seriously IBM's other distributed engines (3790, 8100, System 36, System 38, and various controllers), but at long last we take the 9370 seriously. VM is the tool for integrating preferred guests (IBM's other operating systems--VSE etc.) and for replacing less welcome systems which are on the network--competitive minicomputers whether of external or internal origin.



EXHIBIT V-1

IBM'S PREFERRED SOLUTION





- DB2 is the vehicle for distributing data bases, and it is a good choice. Despite his somewhat abrasive and contentious approach to defending his creation, Dr. Edgar Codd is right about the importance of the attributes of the relational model, especially as they apply to distributed data bases. (Perhaps he will even succeed in getting IBM to shape up after all these years.)
- It is INPUT's opinion that IBM's preferred microprocessor solution will exhibit the same properties as the 9370--370 architecture, VM for integration of existing operating systems, a new proprietary operating system for replacement of existing operating systems (for truly distributed applications), and the common thread of DB2 for distributing data bases.
- It should now be understandable why the flexibility of the IADS which was described in the previous section is so important. There is a great need for providing the flexibility for applications systems (and/or portions of applications systems) to run in different environments. They may either grow up from the desktop or be distributed down from the mainframe. While certain functions can be assigned to their proper level in the processing hierarchy during the design phase of the development cycle, the lesson learned with data bases must be remembered—organizational changes (centralization and decentralization) will occur and flexibility must be maintained to accommodate change in the network structure as well as in the data bases themselves.
- In other words, applications systems are going to continue to evolve in unpredictable ways even if the physical network itself remains constant (which it won't). Fundamentally, the concept of physical data bases will be replaced with one of information flow among and between humans and computers, and the applications systems being developed should be designed with that in mind (more on that later).



- IBM has slowed the development of effective computer/communications networks (distributed processing) despite technological and economic justifications which have been clearly apparent for over a decade. The events of the last year do not indicate that IBM can no longer exercise control over network development--its tools for such control remain awesome (SNA, operating systems, and DBMS)--but they do mean that IBM may choose to exercise a hardware/firmware/software strategy which could be substantially more damaging to certain categories of competitors than it has in the past.

C. MEDIA REVOLUTION

- While the evolution of computer/communications networks has been excruciatingly slow, the move away from paper-based manual systems and the glut of paper information created by current office automation efforts will be substantially more rapid. The problems associated with burgeoning paper glut and the payoffs associated with reducing paper information flow are so great that paper cannot remain the primary information media. In addition, a technological breakthrough in storage media in the form of optical memories is going to make the economics even more compelling.
- Increasing numbers of development centers are going to be adopting the goal that "inefficient manual systems" must be replaced (just as Case Study #3 has already done). The systems being developed and the tools to support such development will be extended beyond the printed document. With optical memory technology being introduced into computer/communications networks at Levels II and III (minicomputers and intelligent workstations) before it is on mainframes, concepts of data base distribution and management are going to have to change significantly.
- IBM has a critical stake in maintaining revenues from magnetic storage while also addressing the more promising aspects of office automation (such as



replacing paper-based systems). It is INPUT's opinion that while IBM may slow the acceptance of optical memories on mainframes, it will be relatively powerless to control their use at the desktop and departmental levels. Critical changes are occurring in the way data/information/knowledge will be stored and communicated, and IBM will not be able to control these changes as they did distributed processing (see Impact of Upcoming Optical Memory Systems, INPUT, 1983, and this year's CD ROM report series for more detailed analysis).

- Considering the fact that paper has been around for a couple of millennia, and the printing press for several hundred years, a fundamental media change away from paper must be considered a revolution. This revolution makes current desktop publishing activities about as important as designing a new kind of buggy whip for a horseless carriage.
- The necessity for maintaining flexibility during times of revolution provides another good argument for maintaining flexibility during the systems development process. Tools and aids should not be tied to existing storage devices, access methods, or media. The media revolution is another good argument for the type of IADS which has been described above.

D. AI AND ALL THAT IMPLIES

- At year end 1986, the usual MIS "wish lists" for 1987 were compiled and published. Practically all of the MIS executives interviewed wanted productivity tools, and many mentioned tools (such as 4GLs) which were already available, immediately raising the question of why they had not done something about their problems already. However, a number responded with statements which indicate they have already adjusted their thinking to bigger and better things. For example:



- One executive of a major bank expressing the need for "one thing and one thing only--error-free code" went on to say: "I want a voice tube that you can speak into and out the bottom comes code."
- Another manager of systems development stated his wish as follows: "I would like to have a piece of software that could design software. With a product like that, I wouldn't need any programmers."
- This type of "fifth generation delusion" is currently being propagated by some advocates of artificial intelligence and expert systems and by many less informed journalists. While this can be expected, it is rather disconcerting to see supposedly responsible MIS executives dreaming of a new generation of magical solutions rather than attending to the problems at hand. Getting rid of programmers is easy, all you have to do is call them analysts, knowledge engineers, end users--practically anything. However, do not expect to eliminate the necessity for communicating with a computer in a logical manner when instructing it what to do, when describing the information you need, and when using its data.
- It should be apparent that the domains of experts systems are entirely too narrow for general purpose problem solving such as that involved in most systems analysis of complex business problems. And if code generation is all that is desired, conventional approaches which are already available will continue to be more productive in the foreseeable future. The probability that AI and/or expert systems will have significant positive impact on productivity in the systems development process within the next ten years is extremely low. In fact, the probability that expert systems will have negative impact on productivity in the systems development process is much more likely as the more creative and inventive people try developing expert systems rather than working on the more practical problems of the business world.
- The big recent news in expert systems is that one vendor has linked its knowledge-based systems with relational data bases. (INPUT suggested this would



be both desirable and necessary in Artificial Intelligence and Expert Systems in 1985.) However, now the controversy within the industry centers on whether such capability should be implementing in "C" rather than Lisp as IntelliCorp has done. Isn't it interesting that some advocates of AI are telling us we should be able to instruct our computers in natural language, but when you build expert systems the language controversy continues to rage?

- As mentioned previously, the trend toward natural languages for instructing computers seems to be something of a misguided effort, primarily because natural language (in our case, English) is inappropriate for humans to use in describing algorithms, procedures, logic, decision trees, data structures, and practically everything else a computer is good at doing. So even if we learn to translate natural language into computer language, what do we have is a language which is difficult to use in describing the very things computers do best.
- The Japanese want to "talk" with their computers for a very good reason--ASCII keyboards just do not hack kanji. It is probably a mistake for the artificial intelligence community to divert a lot of their limited brain power to solving a problem for the Japanese, but some of them seem to be taking the "fifth generation" quite seriously. When natural language replaces LISP for describing the logic (decision rules) in expert systems, we will know the language problem has finally been solved, but you can bet there will be a lot of words in the vocabulary which are not there today, and "open paren" and "close paren" will be the most natural abbreviation in the dialog between man and machine. Who will be kidding whom (man or machine)?
- In fact, I shall here modestly propose the Tim Tyler Test (T3) for determining when natural languages will replace programming languages and when the full promise of artificial intelligence will have been fulfilled. The test is named for an obscure critic of all computer languages which have been developed since APL. The test states that



when it is possible for a human being to dictate his/her job description into a computer and the output becomes an expert system which replaces the job he/she has described, he/she will not care whether the dialog occurs with a computer or a human being. While this test makes the Turing Test somewhat superfluous, it is our opinion that much research in artificial intelligence would remain even if T3 were met. It is for that reason that it is recommended that "knowledge engineers" bootstrap their efforts and describe fully what they are doing so that theirs can be the first profession to be eliminated, thus freeing them to consider the more fundamental philosophical questions of AI while providing computers with a general purpose expert systems generator which could immediately start replacing jobs at a lively clip.

- Enough of that; within a week after the publication of this report, a new company named Virtual Artificial Intelligence will probably announce SUPER SHELL with the Knowledge Engineer Expert Subsystem (SS/KEES) and be looking for venture capital to solve the productivity problem once and for all. Who knows, it might be the best idea anyone has come up with in a long time--kill knowledge engineering in the cradle, and you will never have the problems we have with programming today.
- INPUT is not negative about the efforts which are going on in AI and expert systems. There is a great need for such tools if we are to have any hope of identifying meaningful information and knowledge among the massive amounts of data which are becoming available on computer/communications networks, and the massive amounts of information which are being generated at the human/machine dyad.

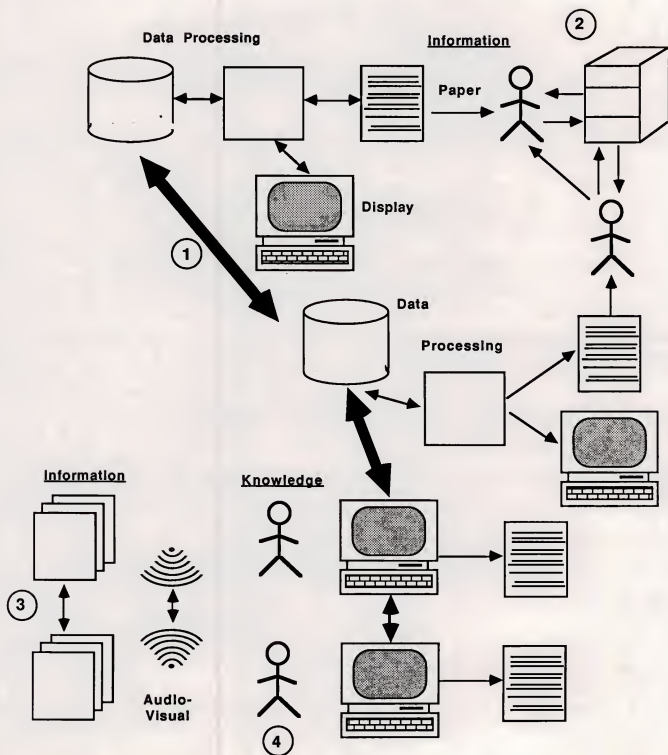


E. THE DATA/INFORMATION/KNOWLEDGE MODEL

- Despite changes of terminology, the major efforts of the past have been to install computer systems and process data. The emphasis of our tools and our systems has been on data capture, data management, and reporting and display of information. It is seldom that the data processing (or information systems) departments have truly concerned themselves with the flow of information in manual systems or between human beings. Only recently has there been any mention of knowledge, much less any concern for how it can be represented in computer systems. More importantly, there has been little concern (much less understanding) of the interplay of data, information, and knowledge.
- The future directions which have been outlined above dictate that emphasis of computer/communications systems must shift away from the data processing/data base orientation which has categorized applications systems development to this point in time. Instead of concentrating on telling computers what to do and how data should be structured and accessed, the emphasis must be on the flow of information and the capture of knowledge. Data processing personnel have been looking at the world through a microscope, and they need to refocus and use a telescope (see Exhibit V-2).
- There is a logical sequence to the refocusing of the development process which is dictated by technology.
 - The first thing which must be considered is the flow of data among computer systems. This is not to be confused with the current emphasis on "connectivity" which is merely the technical connection of diverse computer systems. It is concerned with the distribution of processing and data over the network. Specifically, it is concerned with the quality of data which moves between systems, and when data are combined with programs, the interchange between computers can



THE DATA/INFORMATION/KNOWLEDGE MODEL





be considered information. This is a refinement of the simple definitions of data information and knowledge which were presented earlier in this report.

- The second thing which must be considered is what happens to the paper information which is currently being produced by both computer and manual systems. How does it flow? What is its content? What is its quality? What is its value? How can this information base be managed? Productivity must be measured based on answers to these questions and not the ability to produce paper documents more rapidly and/or more cheaply.
- The third thing which must be considered is how the communication of information and knowledge between and among humans can be improved. It is assumed that electronic systems will be more cost-effective than either paper or audio-visual communications, and the technology to replace much interpersonal communications is now becoming possible. The question once again becomes one of quality and true impact on productivity in the broadest sense. Once again volume of information is not a measure of its value. Some means of measuring utility must be established; overwhelming a human with information does not increase his knowledge or aid the decisionmaking process.
- Finally, the interactions between information and knowledge must be understood. What information is really needed by what individuals? What knowledge resides in which human beings? How can knowledge be extracted from the vast volumes of information which are becoming available?
- Tools and aids to facilitate the control of information flow and its quality are essential in this environment. In addition, the systems should provide for the analysis of information flow in order to gain better understanding of the intricate interplay of information and knowledge. The identification and



capture of knowledge from both human beings and information flow of the artificial system are also essential.

- This terminology is not meant to imply that major technological breakthroughs (such as natural language recognition) are required before substantial progress can be made in these directions. Even good editing facilities are based on knowledge that data should fall within certain ranges, and the knowledge base can be enhanced by discovering that those ranges are not, in fact, correct. Even an enhanced version of the old data processing ploy of asking for reconfirmation that information is still required can be of significant value when built into the information flow.
- The type of IADSs described earlier are coming into existence already and are necessary for facilitating the flexible distribution of processing and data over computer/communications networks.
- Nearly three years ago, in Market Impacts of New Software Productivity Techniques, INPUT outlined a set of tools which would be needed because of the trend toward distributed systems development (DSD). Those tools addressed precisely the type of environment which is today being confirmed by both the recognition of the proper role of mini-computers in the processing hierarchy and the availability of early optical memory systems (CD ROM). Emerging IADS are well suited for the development of those systems.
- Also recommended in the 1984 productivity report was the need for the integration of artificial intelligence and operations research tools; the integration of expert systems with DBMSs was considered to be obvious. The reduction in the amount of paper does not automatically solve the problem of information overload; the problem is only exacerbated on electronic systems. While we consider the development of a general purpose "decision maker" to be unlikely in the foreseeable



future, we believe that a generalized expert system for the screening, classification, and "filing" of information for individuals to be quite possible.

- With information flow being monitored and expert systems becoming knowledgeable of individuals interests and use of information, it is probable that more understanding will be obtained about the interplay between information and knowledge than will be accomplished either in laboratories or within the narrow confines of specific expert system domains.

F. THE USERS' VIEW OF FUTURE PRODUCTIVITY IMPROVEMENT

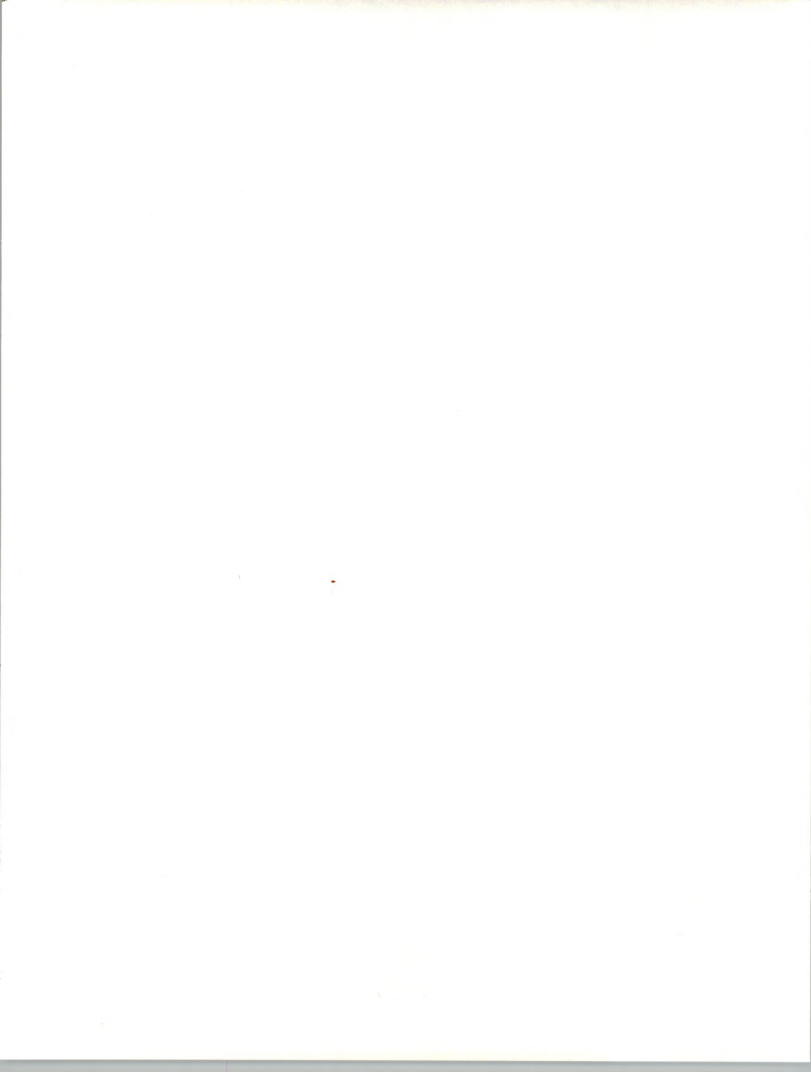
- Users were asked what they considered to be the most likely source of improved productivity in the systems development process, and their responses were classified in a matrix formed from INPUT's productivity hierarchy and performance systems categories. The results are presented in Exhibit V-3.
- Only three gave answers which fell in the commitment to quality category, and two of those were general statements concerning the need for systems developers to become more involved in the company's business problems. The third was a reasonably well defined statement of joint MIS/user project teams (work units) to ensure a quality system.
- Five mentioned increased end-user involvement. These answers demonstrated the continued trend toward the DSD environment, and the emphasis was on prototyping. The split between the human/machine dyad and the work unit was based on our best judgment as to whether the respondent was simply interested in being able to show rapid results (human/machine dyad) or was really interested in forming a work unit with the end user.



EXHIBIT V-3

FUTURE EXPECTATIONS

Performance Level	Commitment To Quality	End User Involvement	Broadbased Management	Effective Personnel	Right Tools
Hardware-Software					7
Human/Machine Dyad		2			12
Work Unit	1	3		4	5
Institutional	2				
Total	3	5	0	4	24



- No users even mentioned management support, much less broadbased management of the development process. This was rather disappointing since INPUT believes that the current productivity problem is primarily one of project management rather than systems design and programming. Nevertheless, it is probably understandable; it is rare to find any systems manager who will really embrace the idea of a planning process. The general attitude is always one of: "...if they only would leave us alone, we would get the job done."

- Four people mentioned that more good people would be the primary way that productivity would be increased. While some put more emphasis on quality than others, the primary concern was a proper level of staffing for the development team. The staffing of development efforts and the impact of bringing people in and/or turnover is an interesting study in its own right. Generally speaking, there comes a time in practically all projects where the loss of critical personnel can impact beyond any reasonable hope of recovery. In fact, adding additional personnel will have a negative impact.

- Not surprisingly, the preponderance of responses (67%) fell in the right tools category. However, there were surprises in the distribution and content of the responses:
 - Seven of the respondents state that more hardware (upgraded mainframes, PCs, connectivity, etc.) is improving systems development productivity. There may be some cases where this is true, but throwing more computer hardware at the problem has not worked very well over the last 30 years.

 - Twelve of the respondents opted for conventional tools (4GLs and DBMS) which are directed primarily at improving individual productivity in implementing systems. It is to be hoped that this



is so, but experience tells us that tools alone will not solve the problem.

- Five respondents mentioned more integrated approaches which are aimed more at the work unit, such as CASE, design methodologies, and project management systems. None of the IADS, mentioned above, received notice in our sample.
- Fundamentally, the solutions users are depending on have not changed appreciably in the last three years. Perhaps that is why INPUT finds that most of its conclusions remain essentially the same, only now we feel they have been substantially reinforced by the changes which have occurred since that time.



VI CONCLUSIONS AND RECOMMENDATIONS





VI CONCLUSIONS AND RECOMMENDATIONS

A. CONCLUSIONS

- Fundamentally, the productivity "problem" is a management problem, pure and simple. Our focus has been too much on computer technology and not the business problems at hand. This criticism is often made of the IS function, but it is important that it be both understood and accepted as having substantial basis in fact. All past INPUT research associated with the productivity hierarchy has revealed the following attitudes on the part of IS management to some degree:
 - Commitment to quality is sacrificed to showing immediate results by getting the system installed and leaving the maintenance problems for someone else. Or, in the case of end-user computing, there has been a general attitude of standing back and waiting for end users to fail (or have to turn back to the IS department for guidance). Prototyping is being used in some organizations to institutionalize quick and dirty systems. Competent IS management knows that all of these attitudes are counterproductive.
 - End-user involvement has only occurred after the end users shot their way into the party with their PCs, and this study confirms the attitude that the primary benefit of distributed systems development (PCs, information centers, prototyping, etc.) is to buffer development teams



from end users. The fortress mentality of "us against them" persists in many IS departments.

- Broadbased management of development projects is not normally accepted as being beneficial—at best it is a necessary evil which slows down the development process. On the other hand, some IS departments seem content to let vendors (most noticeably IBM, but other vendors and consultants as well) set the major directions of development projects by fitting their problems to the vendors' current hardware/software "solutions."
- Effective personnel implies careful selection, motivation, management, and retention of highly qualified employees. Over the years there has been a general tendency to throw bodies at software development problems when every indication is that this does not work and actually has a negative effect. Statistics speak for themselves in the turnover rates of systems personnel and in the floating cadre of IS managers, project leaders, and "executives" who seldom see major projects through before leaving for greener pastures.
- Tools and aids to improve productivity (hardware, software, methodologies, and approaches) have been the focus of so much attention that the more important levels of the productivity hierarchy have virtually been ignored. In many cases, the continuing search for the magical solution has resulted in failure to employ available tools and aids properly.
- While this assessment may seem harsh, there are few IS executives who will not admit that this is the view taken of the IS function by corporate management and end users. Unfortunately, analysis of the software productivity only tends to confirm that these problems literally exist with the IS departments of most organizations.



- The problems associated with productivity in the systems development process are not getting any easier to solve—quite the contrary. Just a few years ago, the systems development environment was closed and relatively manageable. There was a great deal of emphasis on standards and carefully designed systems which would run in the controlled environment of large mainframes. It was fundamentally a one operating system, one DBMS, one language environment for which major applications systems were being developed. The central data processing department created both the applications development environment and the operating environment for the resulting systems.

- This highly centralized, and somewhat comfortable, environment has been changed by technological developments, major shifts in IBM's strategy, and by the advance of the productivity tools which permitted distribution of some applications development activities to end users. These events are briefly summarized as follows:
 - The development of the personal computer destroyed the highly centralized environment and made both distributed processing and distributed systems development (DSD as defined by INPUT) inevitable.

 - IBM has gone from a single, mainstream, systems software environment (MVS and IMS) on large mainframes to a multiple operating system and DBMS strategy by giving VM and DB2 (and the relational model) equal stature. In addition, after years of resisting distributed processing (or at least having a highly suspect strategy), IBM has been forced to face the "connectivity" problem within its own house, and the 9370 announcement would seem to represent IBM's preferred direction for distributed processing. However, it adds additional complexity to the operating environment in terms of systems software.

 - The DSD environment with information centers, prototyping, and PCs has created a new world in which parts of applications (if only reporting) are being written with new tools which are relatively



unfamiliar to the development function (4GLs and a host of PC tools). With different tools being employed on different parts of an applications system and at different levels in the processing hierarchy (which has multiple operating environments), substantial potential for problems exist in terms of systems quality. Specifically, these problems are:

- . Data base integrity and synchronization.
 - . Privacy and security.
 - . Conflicting information (reports) to management.
 - . Increased entropy of data and information which could effectively obscure audit trails and/or meaningful historical information and knowledge.
 - . Hardware/software performance degradation.
- In addition to all of the above, we are on the verge of a major change in media for information storage and communication (from paper to electronic) which promises to have impacts that are more significant and unpredictable than even the personal computer.
- There is no indication that end users can make any substantial contribution to the design or development of the distributed systems which are called for by the changing technological environment. Beyond report generation with word processing systems and spreadsheet packages, most users need help with even the most fundamental system concepts when they start using PC data base systems, and when departmental processors (minicomputers) are installed there is an immediate need for systems personnel (analysts, programmers, or data base administrators). There is no indication from the case studies supporting this research that the "bottom-up" design of applications systems



result in anything (other than perhaps report programs and/or formats) which can ever be effectively integrated with other levels of the processing hierarchy.

- INPUT concludes that the development of distributed applications systems (and data bases) must be carefully designed and controlled if the quality of information is to be maintained (much less improved). Since the computer systems being developed are themselves productivity tools for the enterprise, organization, or institution, it is necessary to measure the productivity of the systems development function on a much broader basis than the ability to generate large quantities of information rapidly. There are four levels of performance which must be considered. Those performance levels, with general conclusions, are as follows:
 - Hardware/Software. The general tendency has been to throw computer hardware/software at problems with the belief that any computer-based system improves productivity regardless of cost. This universal assumption is not warranted, and events of recent years indicate that it is beginning to be questioned. To the degree that our tools and systems development approaches give low priority to hardware/software performance, it is probable that they will come under increasingly severe scrutiny.
 - Human/Machine Dyad. Most productivity improvement has been directed toward this level because performance at this level is theoretically more easy to measure. However, there is little substantiation for any assumption that white collar productivity can be measured by the volume of work produced whether the output is "reports" or computer code.
 - Work Unit. Except for the routine processing of paper documents (in which case the office resembles a factory and can be automated), the product of offices is communication to support business planning,



control, and decision making (even if that decision is which customer to take to lunch). Once again, the effectiveness (productivity) of communications cannot be determined by volume. More time spent in meetings or on the telephone is no clearer an indication of productivity than the volume of paper. Once again the quality of information being communicated is the essential ingredient of true productivity.

- Institutional. The promise of computer systems originally was to save money (usually in clerical and accounting functions) and has since shifted to "making money" through providing a competitive edge. Supporting this competitive edge is the concept that information in itself has value. All too frequently this has been interpreted to mean quantity rather than quality. In addition, the assumption has been made that anything produced by computer is inherently more accurate or of better quality than that produced by other means. There is very little evidence that there is a strong correlation between the amount invested in computerized systems and institutional performance. Indeed, there is every indication that knowledge (people) is substantially more important than information in institutional performance.
- For the above reasons, INPUT has emphasized that commitment to quality is the most important foundation for improved productivity in the systems development process. This means that the IS function must concern itself with the overall flow of information within the organization and not just with the mechanics of producing information from computers. In 1984, INPUT outlined the following set of tools and aids necessary to address the quality of information flow (in Market Impact of New Software Productivity Techniques):
 - An information base management system.
 - A document control system.



- A data flow monitor.
- Improved security, protection, and privacy systems.

The appropriate use of the tools of both operations research and artificial intelligence.

- Events since that time confirm the need for these systems, and it is INPUT's belief that the development of general purpose tools to address these systems requirements will not become readily available from vendors. Even if they do (as in the case of AI shells), substantial effort will be required from the IS department.
- It is INPUT's opinion that IBM will continue to control the essential network, operating systems, and data base environments which exist in the processing hierarchy. Users concerned with the development of applications systems which are distributed over that processing hierarchy must maintain flexibility in order to minimize potential impacts as IBM's systems software strategy evolves and in order to take advantage of specific technological advances which IBM does not support at particular points in time. For these reasons, the Integrated Applications Development System (IADS) described briefly in the body of this report is deemed to be a most pressing requirement for the applications development center. Vendors are beginning to make progress in this area, and as long as such systems are not considered the final solution to the productivity problem, they will materially assist in improving productivity.
- It is also concluded that optical memories will afford an entirely new level of cost justification based on reduction of paper flow. The tools needed to take advantage of this new technology will be more closely related to the library and the fileroom than they are to the data base, but the integration with existing data base management facilities is an obvious requirement which can only be satisfied by the central IS function.



- The emphasis of information systems has been on data base systems which facilitate the capture, structuring, storage, and easy accessibility of data. Impressive tools have been, and are being, developed for these purposes. However, with the shift from paper to electronic media, the enormous amounts of on-line data and information will see a dramatic shift in the requirements for end-user productivity tools. These requirements will emphasize the need for analysis, classification, screening, and communication of information and knowledge on a highly personalized basis. In other words, tools are needed for individuals, work units, and organizations to better manage the product of our current efforts at productivity improvement.
- While these "information management" tools could benefit from many of the concepts of artificial intelligence, it is not necessary to wait for all of the critical problems of artificial intelligence to be solved before useful systems can be developed. Indeed, such systems are the logical extension of the quality assurance systems which were outlined above.
- The impact of such intelligent, personalized tools on applications development will also be favorable because the user will be able to better manage his changing requirements himself. Information will be handled in larger bundles (electronic file folders, archived reports, books, and periodicals). Since information can be scanned by the intelligent aids and browsed through by the individual, the demands on the central IS function for different views of the data (printed reports) will be less severe.
- The concept of intelligent documents has recently surfaced, and while the terminology will probably be as misused as most of our other buzzwords, it is an apt term for the product of the types of quality assurance and intelligent tools which INPUT feels are required. Essentially, an intelligent document should be able to explain its origin (data, programs, and people who created it) and its quality (limitations



and strengths), and it should have the ability to personalize itself for the individual who has requested or received it, screening out certain information of no value and appending (or referencing) information of possible interest.

- In summary, the primary user requirements are no longer for tools to produce information, they are for tools to make intelligent use of information in improving performance at all four levels of the productivity hierarchy--hardware/software, human/machine dyad, work unit, and institutional.

B. RECOMMENDATIONS

- Recognize that the primary purpose of the IS function is to improve productivity at the four performance levels of the performance systems category (hardware/software, human/machine dyad, work unit, and institutional). Understand that the productivity improvement in the systems development process is a subset of the overall productivity improvement problem, but must address all four performance levels.
- Establish a productivity improvement program which addresses the five levels of the productivity hierarchy in priority order--1) commitment to quality, 2) end-user involvement, 3) broadbased management, 4) effective personnel, and, 5) the right tools and aids. This recommendation is neither superficial nor casual; it is repeated in INPUT reports because of our conviction that the software productivity "problem" will only get worse unless the proper emphasis is placed on the management aspects of the systems development process.
- Broaden the focus of the IS department from emphasis on data processing and central data bases to emphasis on quality (as opposed to quantity) of information flow and the process of decision making (including the knowledge required



in that process). This is a fundamental shift of IS emphasis from the production of information to the use of information, and it goes far beyond the changes of terminology (and "lip service") which have occurred over the years.

- Concentrate on establishing the right hardware/software systems environment for your company in anticipation of technological developments and without undue dependency on the hardware/software products of a particular vendor. This implies maintaining flexibility in systems design so that major applications may be properly distributed over computer/communications networks.
- Provide users with the essential tools to assure information quality control and knowledge identification. (These tools are essentially those which have been outlined roughly in past INPUT studies, and performance evaluation as it relates to productivity improvement will be the primary emphasis of the ISP program in 1987.)
- Plan for the shifting role of the IS function from data processing applications developers to systems design consultants for users and general productivity consultants to management.





